Universidad Politécnica de Valencia
Departamento de Ingeniería de Sistemas y Automática



**Tesis Doctoral**

# Control predictivo sujeto a restricciones poliédricas no convexas: Solución explícita y estabilidad.

**Autor:** Emilio Pérez Soler

**Directores:** Dr. Carlos V. Ariño Latorre
Dr. F. Xavier Blasco Ferragud
Dr. Miguel A. Martínez Iranzo

**Valencia, enero de 2011**

*A mis padres y hermanos.*

# Agradecimientos

Esta tesis doctoral es fruto de un importante esfuerzo, pero también es el resultado del apoyo que he recibido de muchas personas.

En primer lugar, me gustaría agradecer a mis directores toda la ayuda que me han brindado a lo largo de los últimos años. Gracias a Xavier y Miguel por introducirme en el mundo de la investigación en general, y del control predictivo en particular. Ambos han compartido sus conocimientos y experiencia, me han ofrecido su confianza y apoyo a lo largo de un extenso período de tiempo y son un referente de esfuerzo y rigor. Gracias también a Carlos por su inestimable ayuda, las incontables horas dedicadas y por transmitirme el entusiasmo y confianza que me han llevado a la finalización de este trabajo.

Gracias al profesor J.A. Rossiter de la universidad de Sheffield por su acogida durante mi estancia y por las discusiones que mantuvimos que, aunque breves, fueron muy esclarecedoras.

Gracias al Gobierno de España por haber financiado durante 48 meses la realización de esta tesis a través del programa de Formación de Personal Investigador (referencia de la ayuda BES-2002-2151).

Me gustaría también dar las gracias a todos los compañeros con los que he coincidido en el Departamento de Ingeniería de Sistemas y Automática de la Universidad Politécnica de Valencia. Han sido muchos a lo largo de estos años pero no quiero dejar de nombrar a Emanuele, Jörn, Alfredo, Sergio, Kiko o Javi, por las horas compartidas, ya sea en el laboratorio, frente a la pantalla del ordenador o delante de un café. Todos ellos han sido un apoyo importante en diferentes momentos de mi etapa en Valencia.

Gracias a mis compañeros de la Universitat Jaume I, con los que comparto ahora todas esas horas. Gracias en particular a José Carlos, Hèctor, Julio, Néstor, Pedro y especialmente a Nacho, por su ayuda con LaTeX. Gracias también a Enrique por su paciencia, y por no preguntar más de lo estrictamente necesario por la fecha de finalización de esta tesis.

A mi familia y amigos de la Venta del Moro. Sin la distracción que han supuesto a lo largo de los años el camino hasta terminar este trabajo posiblemente habría sido más corto, pero con toda seguridad más aburrido. Es imposible nombrarlos a todos, pero gracias por estar ahí todo este tiempo.

Por último, me gustaría agradecer especialmente toda su ayuda a mi familia. A mis padres, Luis y María Ángeles, por haberme dado una educación y ánimos constantes y por demostrarme día a día su confianza en mí. Gracias también a mis hermanos por todo el apoyo que me han brindado y por todo lo que compartimos. A Luis, por ser un ejemplo de sacrificio y superación y a Javi, de quién espero leer en unos años unas líneas como éstas. A ellos cuatro dedico esta tesis.

# Resumen

En esta tesis doctoral se aborda el problema del control predictivo sujeto a restricciones definidas como la unión no convexa de varios poliedros. Los controladores propuestos son de utilidad, por un lado, para procesos que presentan de manera natural restricciones de dicha forma y, por otra parte, como una alternativa al control predictivo no lineal cuando períodos de muestreo bajos no permiten la aplicación de programación no lineal.

En los primeros capítulos del trabajo se demuestra la existencia de una solución explícita a los problemas de optimización que aparecen al plantear este tipo de controladores predictivos. Dicha solución es afín a tramos definidos mediante desigualdades lineales y cuadráticas. Se introducen dos metodologías diferentes para la obtención de esta solución explícita: la metodología de intersección, división y unión y la de la envolvente convexa.
La primera de estas metodologías se basa en formular subproblemas con las restricciones convexas cuya unión forma las restricciones originales y obtener la solución explícita del problema original a partir de las soluciones de dichos subproblemas.
La segunda metodología planteada se basa en el cálculo de la envolvente convexa de los conjuntos de restricciones y la obtención de la solución explícita del problema convexo definido por estas nuevas restricciones. Se demuestra como parte de las regiones de la solución explícita del problema original coinciden con las del nuevo problema, y se propone un procedimiento para identificarlas y obtener el resto de regiones, completando la solución explícita buscada.

Se estudian también algoritmos eficientes para la implementación en línea de leyes de control explícitas como las obtenidas. En particular, se propone un algoritmo basado en un árbol binario de una partición lineal y una comparación de índices de costes en las regiones en las que sea necesario. Se compara el coste computacional del algoritmo con el de la implementación de un árbol binario para cada uno de los subproblemas convexos, demostrándose que el primero es siempre inferior.

Considerando el caso en el que un período de muestreo bajo impida la aplicación de los algoritmos propuestos por su coste en línea, se proponen dos soluciones subóptimas, una basada en la simplificación del problema de optimización y otra en la simplificación de la solución explícita.

Por otro lado, se analiza también la estabilidad de los sistemas de control propuestos, particularizando al caso de restricciones poliédricas no convexas las condiciones generales de estabilidad *a priori* existentes para esquemas de control con horizonte móvil. Para ello, se propone un algoritmo eficiente para calcular el máximo conjunto invariante en bucle cerrado basado en obtener dicho conjunto para un problema definido por las envolventes convexas de las restricciones del que se eliminan convenientemente algunos estados. Además, se demuestra que para el problema planteado el índice de coste es continuo, lo que permite asegurar que, si se cumplen todas las condiciones de estabilidad, el sistema en bucle cerrado no es únicamente estable en el sentido de Lyapunov, sino asintóticamente estable.

Por último, en la tesis se plantean diferentes problemas en los que son de utilidad las técnicas propuestas. En primer lugar, el problema de evitación de obstáculos y planificación de trayectorias, que en el caso general puede presentar restricciones poliédricas no convexas. En segundo lugar, se propone la aplicación de las técnicas planteadas al problema del control predictivo con restricciones no lineales. En particular, se abordan de este modo el control predictivo de sistemas Hammerstein-Wiener con restricciones mediante cancelación de las no linealidades estáticas y el control predictivo de sistemas no lineales con restricciones mediante linealización por realimentación de entrada-salida.

# Resum

En aquesta tesi doctoral s'aborda el problema del control predictiu subjecte a restriccions definides com la unió no convexa de diversos poliedres. Els controladors proposats són d'utilitat, d'una banda, per a processos que presenten de manera natural restriccions d'aquesta forma i, d'altra banda, com una alternativa al control predictiu no lineal quan períodes de mostreig baixos no permeten l'aplicació de programació no lineal.

En els primers capítols del treball es demostra l'existència d'una solució explícita als problemes d'optimització que apareixen en plantejar aquest tipus de controladors predictius. Aquesta solució és afí a trams definits mitjançant desigualtats lineals i quadràtiques. S'introdueixen dues metodologies diferents per a l'obtenció d'aquesta solució explícita: la metodologia d'intersecció, divisió i unió i la de l'envolupant convexa.
La primera d'aquestes metodologies es basa a formular subproblemes amb les restriccions convexes la unió de les quals forma les restriccions originals i obtenir la solució explícita del problema original a partir de les solucions d'aquests subproblemes.
La segona metodologia plantejada es basa en el càlcul de l'envolupant convexa dels conjunts de restriccions i l'obtenció de la solució explícita del problema convex definit per aquestes noves restriccions. Es demostra com a part de les regions de la solució explícita del problema original coincideixen amb les del nou problema, i es proposa un procediment per a identificar-les i obtenir la resta de regions, completant la solució explícita cercada.

S'estudien també algorismes eficients per a la implementació en línia de lleis de control explícites com les obtingudes. En particular, es proposa un algorisme basat en un arbre binari d'una partició lineal i una comparació d'índexs de costos en les regions en les quals siga necessari. Es compara el cost computacional de l'algorisme amb el de la implementació d'un arbre binari per a cadascun dels subproblemes convexs, demostrant-se que el primer és sempre inferior.

Considerant el cas en el qual un període de mostreig baix impedisca l'aplicació dels algorismes proposats pel seu cost en línia, es proposen dues solucions subòptimes, una basada en la simplificació del problema d'optimització i una altra en la simplificació de la solució explícita.

D'altra banda, s'analitza també l'estabilitat dels sistemes de control pro-

posats, adaptant al cas de restriccions polièdriques no convexes les condicions generals d'estabilitat *a priori* existents per a esquemes de control amb horitzó mòbil. Per a açò, es proposa un algorisme eficient per a calcular el màxim conjunt invariant en bucle tancat basat a obtenir aquest conjunt per a un problema definit per les envolupants convexes de les restriccions del que s'eliminen convenientment alguns estats. A més, es demostra que per al problema plantejat l'índex de cost és continu, la qual cosa permet assegurar que, si es compleixen totes les condicions d'estabilitat, el sistema en bucle tancat no és únicament estable en el sentit de Lyapunov, sinó asimptoticament estable.

Finalment, en la tesi es plantegen diferents problemes en els quals són d'utilitat les tècniques propostes. En primer lloc, el problema d'evitació d'obstacles i planificació de trajectòries, que en el cas general pot presentar restriccions polièdriques no convexes. En segon lloc, es proposa l'aplicació de les tècniques plantejades al problema del control predictiu amb restriccions no lineals. En particular, s'aborden d'aquesta manera el control predictiu de sistemes Hammerstein-Wiener amb restriccions mitjançant cancelůlació de les no linealitats estàtiques i el control predictiu de sistemes no lineals amb restriccions mitjançant linealització per realimentació d'entrada-eixida.

# Abstract

In this thesis, it is addressed the problem of predictive control of linear systems subject to non-convex polyhedral constraints, defined as the non-convex union of several convex polyhedra. The proposed controllers are useful, first, in problems for which such constraints arise naturally and, second, as an alternative to non-linear predictive control when low sampling times don't allow the use of non-linear programming.

In the first chapters of the work, it is shown the existence of an explicit solution to the optimization problems that arise when this kind of predictive controllers is proposed. Such explicit solution is shown to be piecewise affine defined over regions described by linear and quadratic inequalities. Two different methodologies are proposed in order to obtain this explicit solution: an intersection, division and union methodology and a convex hull methodology.
The former of these methodologies is based on formulating several subproblems with convex polyhedral constraints which union forms the original constraints and obtaining the explicit solution to the original problem from the solutions to these subproblems.
The latter is based on calculating the convex hull of the non-convex constraints and obtaining the explicit solution to the convex problem defined with these new constraints. Next, tt is shown tnat a subset of the regions of the explicit solution to the original problem are equal to some of the regions of the solution to this new problem, and a procedure to identify them and obtain the rest of regions is proposed, finding this way the complete explicit solution searched.

Efficient algorithms for the online implementation of explicit control laws in the previously described form are also studied. Particularly, an algorithm based on a binary tree of the linear partition and a comparison of the cost functions when required, is proposed. The computational burden of such an algorithm is compared to that obtained by the implementation of a binary tree for each of the convex subproblems, showing that the former is always lower than the latter.

Considering the case for which a low sampling time prevents the use of the proposed algorithms because of their online computational cost, two suboptimal solutions are proposed. The first one is based on a simplification of the optimization problem, while the second one is based on the simplification of

the explicit solution.

Furthermore, stability of the control systems proposed is analyzed, specifying the well-known *a priori* stability conditions for general receding horizon schemes to the case of systems with non-convex polyhedral constraints. To do so, a novel efficient algorithm for the computation of the closed-loop maximal admissible set for this kind of systems is proposed. This algorithm starts from the maximal admissible set of the problem with constraints defined as the convex hull of the original ones, and then eliminates specific subsets of it. Furthermore, continuity of the cost function for the proposed problem is shown which allows to prove that, if the stability conditions hold, the system is not only Lyapunov stable but also asymptotically stable.

Finally, some problems for which the techniques proposed in the thesis are useful are introduced. First, the obstacle avoidance and path planning problem is introduced, showing that, for a general case, non-convex polyhedral constraints arise. Second, the application of the developed techniques to the problem of predictive control with non-linear constraints is proposed. Particularly, predictive control of Hammerstein-Wiener systems with static non-linearities cancelation and input-output feedback linearization schemes for predictive control of non-linear systems are solved with our approach.

# Índice de contenidos

Capítulo 1

# Introducción

## 1.1. Motivación

El control predictivo ha demostrado su potencial y ha sido ampliamente aceptado en el control de procesos multivariables a nivel industrial, especialmente debido a su capacidad de incorporar en el diseño el tratamiento de restricciones. Por ello es objeto de estudio a nivel científico en el área de control de procesos, presentando un continuo desarrollo. Su espectro de aplicación se ha extendido a todo tipo de procesos pudiendo considerase como extremos por un lado el control predictivo no lineal sujeto a restricciones no lineales, y por el otro el control predictivo lineal con restricciones poliédricas. En el primer caso, es necesario recurrir a algoritmos en línea de optimización no lineal (y en general no convexa) bastante costosos computacionalmente por lo que sólo podrá ser aplicado a procesos que admitan períodos de muestreo relativamente altos. En el extremo opuesto, el control de procesos lineales sujetos a restricciones poliédricas, existen diversos algoritmos eficientes en la literatura, destacando especialmente [BMDP02] en el que se obtiene fuera de línea una solución explícita definida sobre regiones poliédricas. Existen además técnicas en la literatura que permiten reducir considerablemente el tiempo computacional en línea de dicha solución [TJB03b], haciendo la filosofía de control predictivo factible para procesos considerablemente rápidos.

Si se desea trabajar con períodos de muestreo relativamente bajos, una generalización del control predictivo lineal que permite ampliar su aplicabilidad a diferentes problemas se consigue utilizando modelos lineales con restricciones poliédricas no convexas, definidas como la unión no convexa de un número finito de poliedros. Este tipo de restricciones aparecen de forma natural en problemas como el de evitación de obstáculos, inherentemente no convexo.

Otro campo de aplicación en el que aparecen restricciones no convexas surge a partir de los sistemas no lineales con restricciones. Dentro de este ámbito se encuentran los sistemas de tipo Hammerstein-Wiener, definidos por no lineali-

dades estáticas que siguen y preceden a un modelo lineal. Estos modelos suelen controlarse mediante la inversión de las no linealidades, transformando el problema en uno de control de un proceso lineal, controlable mediante técnicas lineales estándar, por ejemplo control predictivo. No obstante, cuando existen restricciones sobre las entradas y salidas, deben ser transformadas también por la función no lineal para trasladarlas al espacio de variables en el que trabaja el controlador, pudiendo originar regiones no convexas. En este caso, puede conseguirse una aproximación interior suficientemente precisa mediante el uso de varios poliedros cuya unión no es convexa.

Otro problema del mismo tipo también puede aparecer en sistemas no lineales con restricciones controlados mediante linealización de entrada-salida por realimentación. Dichas restricciones, aunque sean lineales, se transforman en no lineales para el modelo linealizado, por lo que, si son no convexas, de nuevo pueden ser aproximadas mediante unión no convexa de poliedros.

## 1.2. Objetivos

El objetivo principal de la tesis es contribuir al avance en el control predictivo de sistemas con restricciones poliédricas no convexas. En particular, se pretende que las técnicas de control propuestas en esta tesis constituyan una alternativa de control predictivo no lineal para algunos tipos de sistemas cuando los períodos de muestreo utilizados en la implementación del controlador sean relativamente bajos. Para ello, se puede concretar el objetivo de la tesis en tres áreas diferentes:

- Eficiencia computacional del algoritmo en línea.
- Estabilidad del sistema en bucle cerrado.
- Aplicación a esquemas de control predictivo no lineal.

### 1.2.1. Eficiencia computacional del algoritmo en línea

Por la naturaleza del control predictivo, que requiere de la resolución de un problema de optimización para el cálculo de la acción de control a aplicar, la consideración de restricciones no convexas y períodos de muestreo bajos supone una dificultad fundamental de partida: la optimización que es necesario resolver es no convexa y los métodos habitualmente utilizados para ello son costosos computacionalmente. Se buscarán por tanto alternativas a la resolución de dicho problema en línea.

Con este objetivo, en analogía con el caso de sistemas con restricciones poliédricas, se pretende estudiar la existencia de una ley de control explícita solución al problema planteado, así como su caracterización. Lo que se desea conseguir con ello es trasladar fuera de línea una parte importante de la carga

computacional requerida para la optimización. Se buscará además la máxima simplicidad posible en la definición de la solución explícita, principalmente haciendo mínimo el número de regiones en los que se particiona el espacio de estados.

También en el ámbito de la eficiencia computacional en línea, se analizará la mejor manera de implementar las leyes de control explícitas que se obtengan como solución al problema. Para ello, se compararán diferentes algoritmos usando como parámetro fundamental el número de operaciones elementales a realizar en el peor caso.

Aunque los algoritmos en línea estén diseñados para reducir el coste computacional requerido, para algunos casos es posible que, por la complejidad del problema, la solución que se obtenga no pueda implementarse de manera suficientemente rápida para el período de muestreo deseado. Para estos casos, se propondrán soluciones subóptimas que reduzcan la carga computacional, a costa de obtener una acción de control no óptima.

Debido al interés por la aplicabilidad a procesos rápidos de las técnicas de control que se proponen en la tesis, el mayor énfasis al estudiar diferentes alternativas se hará en el tiempo de cálculo en línea. No obstante, también se considerarán otros dos aspectos que pueden limitar la utilidad de las técnicas propuestas: la memoria necesaria para el algoritmo en línea y el coste computacional del algoritmo fuera de línea.

### 1.2.2. Estabilidad del sistema en bucle cerrado

Una de las propiedades fundamentales que cabe exigir a un determinado sistema de control es que el sistema en bucle cerrado resultante de aplicarlo al proceso para el que se diseña sea estable, bien en el sentido de Lyapunov o, idealmente, asintóticamente estable.

Para un esquemas de control predictivo genérico, es un resultado conocido que se pueden conseguir garantías de estabilidad mediante la utilización de una combinación de una ley de control, función de coste y región terminales que deben cumplir ciertas condiciones [MRRS00].

Será un objetivo de la tesis diseñar estos elementos de forma que los controladores predictivos planteados sean estables. Además, con un correcto diseño de los elementos enumerados, especialmente la región terminal, se pretende minimizar el incremento de la complejidad del algoritmo de control en línea que supone garantizar la estabilidad del sistema.

### 1.2.3. Aplicación a esquemas de control predictivo no lineal

Si bien existen problemas que por su propia naturaleza presentan restricciones poliédricas no convexas, se pretende aumentar la aplicabilidad de los resultados de la tesis como alternativa de control predictivo no lineal.

Debido a la complejidad de la programación no lineal, es una práctica habitual del control predictivo no lineal evitar su utilización cuando esto es posible. Para ello, lo más habitual es restringir el modelo no lineal que se usa como modelo de predicción a una estructura determinada y aprovechar alguna particularidad de ésta. En concreto, los sistemas de Hammerstein-Wiener y los sistemas linealizables por realimentación son manipulables de diferentes maneras para conseguir transformar el problema de optimización no lineal en uno más sencillo (habitualmente programación cuadrática). No obstante, en ambos casos esto introduce dificultades en el manejo de las restricciones, que en general dejan de ser lineales.

El último objetivo de la tesis es implementar en este tipo de procesos el control predictivo con restricciones poliédricas no convexas.

## 1.3. Estructura de la tesis

Los diferentes temas abordados en la tesis se estructuran en siete capítulos, además de esta introducción.

En el capítulo 2 se introduce formalmente el control predictivo y se presentan las aportaciones bibliográficas fundamentales existentes en el ámbito de la obtención y cálculo en línea de la solución explícita para problemas lineales. Además, se revisa el estado del arte para el control predictivo no lineal y se enuncian las condiciones de estabilidad para este tipo de controladores. Por último, se presentan algunas herramientas matemáticas que serán utilizadas en capítulos posteriores de la tesis.

En el capítulo 3 se introduce el problema a resolver y se muestra la existencia de una solución explícita afín a tramos.

En el capítulo 4 se plantean y comparan diferentes metodologías para la obtención de la solución explícita del problema.

En el capítulo 5 se desarrollan algoritmos en línea eficientes, comparando de manera teórica el coste computacional y los requerimientos de memoria para cada uno de ellos. Además, se plantean soluciones subóptimas al problema planteado.

En el capítulo 6 se revisan las condiciones de estabilidad de Lyapunov y asintótica para esquemas de control predictivo, se particularizan dichas condiciones para el problema con restricciones poliédricas no convexas y se plantean algoritmos eficientes para la obtención de la región de restricciones terminal que permite el cumplimiento de las condiciones propuestas.

En el capítulo 7 se muestran las aplicaciones de las técnicas desarrolladas, dividiéndolas en dos grandes grupos: problemas con restricciones poliédricas no convexas puras y problemas con restricciones no lineales. Para cada una de las posibles aplicaciones se resuelven ejemplos de aplicación.

Por último, en el capítulo 8 se resumen las conclusiones de la tesis y se proponen algunas líneas de investigación futuras.

Capítulo 2

# Estado del arte

## 2.1. Introducción

El control predictivo es un técnica ampliamente extendida en el control de procesos multivariables, especialmente cuando éstos están sometidos a restricciones. En este capítulo se introduce formalmente esta técnica de control y se realiza una revisión del estado del arte haciendo hincapié en tres temáticas fundamentales: la obtención de soluciones explícitas de realimentación del estado, que permiten el diseño de algoritmos en línea eficientes; las técnicas de control predictivo no lineal, en especial el control de sistemas afines a tramos; y la estabilidad en bucle cerrado de sistemas con controladores predictivos.

Además, en este capítulo se introducen algunas herramientas matemáticas que se emplearán en capítulos posteriores. Por un lado, se presentan algoritmos existentes para la realización de dos operaciones sobre poliedros: obtención de la envolvente convexa y minimización del número de elementos cuya unión define un poliedro no convexo. Por otro, se describen las técnicas existentes para comprobar la factibilidad de sistemas de ecuaciones y desigualdades, tanto si éstas son lineales como polinómicas.

## 2.2. Control predictivo

Control predictivo es un término que engloba toda una serie de algoritmos de control que comparten una misma filosofía. En dichos algoritmos, siempre están presentes los siguientes elementos [CB04, Ros03]:

- Modelo de predicción. Se utiliza de forma explícita para predecir la salida del proceso en instantes de tiempo futuros.
- Índice de coste. Representa una función objetivo, que se desea minimizar.

- Optimizador. Obtiene las acciones de control óptimas para el índice de coste propuesto.
- Horizonte móvil. Consiste en que en cada instante de muestreo sólo se aplica la primera acción de control calculada.

Por tanto, todo control predictivo implica la resolución de un problema de optimización en bucle abierto y la aplicación del principio de horizonte móvil. De una manera genérica, el problema de optimización se puede formular:

$$\begin{aligned}\mathcal{P}_{N,M}(x): \quad & V_N^{OPT}(x) := \min V_N(\{x_k\},\{u_k\}), \\ & \text{sujeto a:} \\ & x_{k+1} = f(x_k, u_k) \text{ para } k = 0,\dots,N-1, \\ & x_0 = x, \\ & u_k \in \mathbb{U} \text{ para } k = 0,\dots,M-1, \\ & u_k = u_{M-1} \text{ para } k = M,\dots,N-1, \\ & x_k \in \mathbb{X} \text{ para } k = 0,\dots,N, \\ & x_N \in \mathbb{X}_f \subset \mathbb{X}, \\ & V_N(\{x_k\},\{u_k\}) := F(x_N) + \textstyle\sum_{k=0}^{N-1} L(x_k, u_k)\end{aligned} \tag{2.1}$$

donde:

- $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ es una función dada, en general no lineal, que describe el comportamiento dinámico del sistema.
- $N$ y $M$ son los horizontes de predicción y de control, respectivamente.
- $V_N(\{x_k\},\{u_k\})$ es la función de coste, siendo las funciones $F$ y $L$ la *penalización terminal del estado* y *penalización del estado* respectivamente.
- $\{x_0,\dots,x_N\}$, $x_k \in \mathbb{R}^n$, y $\{u_0,\dots,u_{N-1}\}$, $u_k \in \mathbb{R}^m$, son las secuencias de los vectores de estados y de acciones de control respectivamente.
- $x$ es el estado actual medido del sistema.
- $\mathbb{U} \subset \mathbb{R}^m$,$\mathbb{X} \subset \mathbb{R}^n$ y $\mathbb{X}_f \subset \mathbb{R}^n$ son conjuntos de restricciones.

La solución a dicha problema es una secuencia de control que minimiza la función de coste:

$$\mathscr{U}_x^{\text{opt}} := \{u_0^{\text{opt}}, u_1^{\text{opt}}, \dots, u_{N-1}^{\text{opt}}\} \tag{2.2}$$

Donde el subíndice $x$ denota la dependencia de dicha secuencia del estado actual.

### 2.2.1. Control predictivo de procesos lineales

Hasta el momento, se ha descrito el problema de una manera muy genérica. Si se particulariza, es posible resolver dicho problema utilizando algoritmos de optimización sencillos, además de asegurar ciertas propiedades de la solución que serán de utilidad. Para ello, se realizan las siguientes suposiciones:

- El sistema es lineal e invariante en el tiempo y está descrito por el siguiente modelo:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k, \\ y_k &= Cx_k, \end{aligned} \tag{2.3}$$

- El sistema $(A, B, C)$ anterior es estabilizable y detectable.

- Las restricciones $\mathbb{U}$,$\mathbb{X}$ y $\mathbb{X}_f$, son poliédricas, definidas mediante desigualdades lineales.

- El índice de coste se define para el seguimiento de una determinada referencia, $y^*$, mediante la siguiente función cuadrática:

$$V_{N,M}(\{x_k\}, \{u_k\}, \{y_k\}) := F(x_N - x_s) + \frac{1}{2}\sum_{k=0}^{N-1}(y_k - y^*)^T Q(y_k - y^*) + \\ + \frac{1}{2}\sum_{k=0}^{M-1}(u_k - u_s)^T R(u_k - u_s)$$

  donde:

  - $P \succeq 0$,$Q \succeq 0$ y $R \succ 0$.
  - $u_s$ y $x_s$ son los valores deseados en régimen permanente para la entrada y el estado respectivamente y pueden calcularse a partir de la referencia $y^*$ como:

$$\begin{aligned} u_s &= (C(I - A)^{-1}B)^{-1}y^* \\ x_s &= (I - A)^{-1}Bu_s \end{aligned}$$

Los valores en régimen permanente $u_s$ y $x_s$ sólo podrán obtenerse si se cumple:

- El número de salidas es igual al número de entradas.

- $(I - A)$ es invertible.

- $C(I - A)^{-1}B$ también es invertible.

Con objeto de simplificar la nomenclatura, y sin pérdida de generalidad, en adelante se supondrá $y^* = 0$, $u_s = 0$ y $x_s = 0$. De esta forma, el problema de

optimización a resolver queda como:

$$\mathcal{P}_{N,M}(x) : V_{N,M}^{OPT}(x) := \min\ F(x_N) + \frac{1}{2}\sum_{k=0}^{N-1} x_k^T C^T Q C x_k + \frac{1}{2}\sum_{k=0}^{M-1} u_k^T R u_k \tag{2.4}$$

sujeto a:
$x_{k+1} = Ax_k + Bu_k$ para $k = 0, \ldots, N-1$,
$x_0 = x$,
$u_k \in \mathbb{U}$ para $k = 0, \ldots, M-1$,
$u_k = 0$ para $k = M, \ldots, N-1$,
$x_k \in \mathbb{X}$ para $k = 0, \ldots, N$,
$x_N \in \mathbb{X}_f \subset \mathbb{X}$

Como se discutirá posteriormente, con objeto de conseguir la estabilidad del sistema en bucle cerrado, la penalización terminal del estado, $F(x_N)$, se escoge como la función de coste del problema de control óptimo con horizonte infinito sin restricciones del sistema:

$$F(x_N) = \frac{1}{2} x_N^T P x_N$$

donde $P$ es la solución a una ecuación algebraica de Riccati.

Aplicando consecutivamente la ecuación del modelo lineal del sistema (2.3), es posible reescribir el índice de coste (2.4) como:

$$V_{N,M}(x) = \frac{1}{2}\mathbf{u}^T H \mathbf{u} + \mathbf{u}^T F x$$

donde $\mathbf{u} := [u_0\ u_1\ \ldots\ u_{M-1}]^T$ contiene la secuencia de acciones de control con M movimientos y las matrices $H$ y $F$ se calculan como:

$$H = \Gamma^T \mathbf{Q} \Gamma + \mathbf{R} \tag{2.5}$$
$$F = \Gamma^T \mathbf{Q} \Omega \tag{2.6}$$

donde:

$$\Gamma := \begin{bmatrix} B & 0 & \ldots & 0 & 0 \\ AB & B & \ldots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ A^{M-1}B & A^{M-2}B & \ldots & AB & B \\ A^M B & A^{M-1}B & \ldots & A^2 B & AB \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ A^{N-1}B & A^{N-2}B & \ldots & \ldots & A^{N-M}B \end{bmatrix}, \quad \Omega := \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^N \end{bmatrix} \tag{2.7}$$

$$\mathbf{Q} := \mathrm{diag}\{C^T Q C, \ldots, C^T Q C, P\}$$
$$\mathbf{R} := \mathrm{diag}\{R, \ldots, R\}$$

En cuanto a los conjuntos de restricciones $\mathbb{U}$,$\mathbb{X}$ y $\mathbb{X}_f$, dado que son todos poliédricos, haciendo uso de nuevo de la ecuación (2.3) es posible formularlos como desigualdades sobre las acciones de control [GS05]:

$$\Phi \mathbf{u} \leq \Delta - \Lambda x \tag{2.8}$$

Por tanto, el problema de optimización que se desea resolver es:

$$\begin{aligned} \mathcal{P}_{N,M}(x): \quad & V_{N,M}^{OPT}(x) := \min \tfrac{1}{2}\mathbf{u}^T H \mathbf{u} + \mathbf{u}^T F x, \\ & \text{sujeto a:} \\ & \Phi \mathbf{u} \leq \Delta - \Lambda x \end{aligned} \tag{2.9}$$

La elección realizada de las matrices de penalización $R$,$Q$ y $P$ y de los conjuntos de restricciones $\mathbb{U}$,$\mathbb{X}$ y $\mathbb{X}_f$, implican que el problema pueda ser formulado y resuelto como un problema de programación cuadrática (QP) convexo. Nótese en particular que, por un lado, al ser las matrices $Q$ y $P$ semidefinidas positivas, $\mathbf{Q}$ también lo es y por el otro, la elección de $R \succ 0$ implica que $\mathbf{R}$ sea también definida positiva. Por tanto, la matriz $H$ será definida positiva.

### 2.2.2. Horizonte móvil

Una vez resuelto el problema de optimización, el control predictivo aplica el principio de horizonte móvil, que se puede resumir de la siguiente manera:

1. En el instante $i$, y para el estado actual $x_i$, se formula un problema de optimización sobre un intervalo futuro fijo, $[i, i+N-1]$.

2. De la secuencia de control obtenida (2.2), se aplica únicamente el primer paso, $u_0^{\text{opt}}$.

3. Se mide el estado que alcanza el sistema en el instante $i+1$.

4. Se vuelve a resolver el problema de optimización (2.1), esta vez para el intervalo fijo $[i+1, i+N]$ y para el nuevo estado actual, $x_{i+1}$.

El procedimiento de horizonte móvil descrito, define implícitamente una ley de control $\mathcal{K}_{N,M} : \mathbb{X} \to \mathbb{U}$ de la forma:

$$\mathcal{K}_{N,M}(x) = u_0^{\text{opt}} \tag{2.10}$$

Si se mantienen los horizonte $N$ y $M$ fijos, y el modelo y la función de coste son invariantes en el tiempo, puede observarse que la acción de control obtenida mediante (2.10) será siempre la misma para un mismo valor del estado. Por tanto, el principio de horizonte móvil produce una ley de control invariante en el tiempo.

## 2.3. Control predictivo explícito

### 2.3.1. Objetivos

Como se ha visto, en general cualquier control predictivo puede calcularse resolviendo en línea un problema de optimización asociado. Esto proporciona una ley de control de forma implícita. No obstante, en determinadas circunstancias, es posible obtener una formulación explícita de la ley de control que ofrece ciertas ventajas respecto a la formulación implícita. El objetivo de esta sección es describir como puede obtenerse dicha formulación.
En la bibliografía existen diferentes propuestas que muestran la formulación explícita del control predictivo. Una de ellas es el enfoque geométrico de [GS05], que se muestra en el siguiente epígrafe. Mediante dicho enfoque se analiza, desde el punto de vista geométrico,el problema de optimización asociado a todo controlador predictivo y se utilizan estos conceptos geométricos para obtener la caracterización explícita de la solución.
El mismo resultado puede conseguirse mediante el enfoque de [BMDP02] mostrado en la sección 2.3.4, donde se formula el control predictivo como un problema de optimización cuadrática multiparamétrica y, mediante las condiciones de optimalidad de Karush-Kuhn-Tucker (KKT), se demuestra que la solución a dicho problema se puede expresar como una función afín a tramos.

### 2.3.2. Geometría de la programación cuadrática

El problema de programación cuadrática planteado (2.9), puede interpretarse fácilmente desde un punto de vista geométrico en el espacio de las acciones de control $\mathbf{u}$. Para ello, se considera la ecuación de las curvas de nivel, es decir, el conjunto de valores de $\mathbf{u}$ para los que el índice de coste toma un cierto valor constante $c$:

$$\frac{1}{2}\mathbf{u}^T H \mathbf{u} + \mathbf{u}^T F x = c \tag{2.11}$$

La ecuación (2.11) es la que define una hipercuádrica en $\mathbb{R}^{Mm}$, cuya forma depende del Hessiano (matriz $H$). En nuestro caso, $H$ es definida positiva, por lo que las curvas de nivel son en realidad hiperelipsoides con centro en el óptimo sin restricciones $\mathbf{u}_{uc}^{opt} = -H^{-1}Fx$. El problema de optimización 2.9 puede ser por tanto entendido de la siguiente forma: *encontrar el punto en el que el menor hiperelipsoide con centro en* $\mathbf{u}_{uc}^{opt}$ *intersecta al espacio de restricciones definido por* (2.8).

**Ejemplo 2.1.** *Para un determinado valor del estado,* $x = [1 \quad 1]^T$*, se desea resolver el problema de optimización* (2.9) *en dos variables* ($Mm = 2$) *definido por las siguientes matrices:*

$$H = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} \quad F = \begin{bmatrix} -1 & 0 \\ 0 & -3 \end{bmatrix}$$

*Además, existen restricciones sobre las acciones de control:*

$$\begin{bmatrix} -1 \\ -1 \end{bmatrix} \leq \mathbf{u} \leq \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

*En la figura 2.1 se representan las curvas equipotenciales* (2.11)*, centradas en el óptimo sin restricciones* $\mathbf{u}_{uc}^{opt} = -H^{-1}Fx = [-1\ 2]^T$*: Se observa que el óptimo*

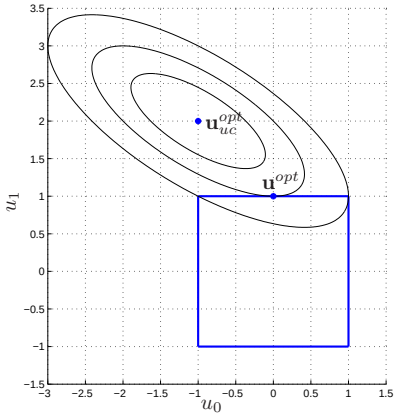


***Figura 2.1:*** Problema de optimización. Interpretación gráfica.

*solución a nuestro problema es aquel que se obtiene de intersectar la frontera del espacio de restricciones con la menor elipse posible:*

$$\mathbf{u}^{opt} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

■

Aunque conceptualmente la obtención del óptimo con restricciones es muy sencilla, a priori no es evidente calcular en qué punto se alcanzará el óptimo, ni siquiera saber exactamente sobre cuál de las restricciones estará. No obstante, el problema puede simplificarse realizando un cambio de coordenadas mediante la raíz cuadrada del Hessiano:

$$\widetilde{\mathbf{u}} = H^{1/2}\mathbf{u} \tag{2.12}$$

Sustituyendo (2.12) en (2.9), el problema de optimización con la nueva formulación es:

$$\mathcal{P}_{N,M}(x): \qquad V_{N,M}^{OPT}(x) := \min \frac{1}{2}\widetilde{\mathbf{u}}^T\widetilde{\mathbf{u}} + \widetilde{\mathbf{u}}^T H^{-1/2}Fx, \tag{2.13}$$

sujeto a:

$$\widetilde{\Phi}\widetilde{\mathbf{u}} \leq \Delta - \Lambda x \tag{2.14}$$

donde

$$\widetilde{\Phi} = \Phi H^{-1/2}$$

Por tanto, se han modificado tanto el índice a optimizar como el espacio de restricciones. La ventaja fundamental de formular el problema de esta forma es que el nuevo Hessiano es la matriz identidad y, por lo tanto, las curvas de nivel (2.11) son hiperesferas, en lugar de hiperelipsoides. Así, resolver el problema de optimización en este caso equivaldrá a *encontrar el punto dentro del espacio de restricciones más próximo en distancia Euclídea al óptimo sin restricciones* $\mathbf{u}_{uc}^{opt}$, o lo que es lo mismo, obtener la *proyección de* $\mathbf{u}_{uc}^{opt}$ *sobre el espacio de restricciones.*

**Ejemplo 2.2.** *Realizando el cambio de variable* (2.12) *al problema propuesto en el ejemplo 2.1, se obtiene un problema de la forma* (2.13)*, cuyo óptimo sin restricciones viene dado por:*

$$\widetilde{\mathbf{u}}_{uc}^{opt} = -H^{-1/2}Fx = \begin{bmatrix} 0 \\ 2{,}2361 \end{bmatrix} \tag{2.15}$$

*Por otro lado, las restricciones tras el cambio son:*

$$\begin{bmatrix} 1{,}341 & -0{,}447 \\ -0{,}44721 & 0{,}894 \\ -1{,}341 & 0{,}447 \\ 0{,}447 & -0{,}894 \end{bmatrix} \widetilde{\mathbf{u}} \leq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

*Las curvas de nivel para el nuevo problema se muestran en la figura 2.2.*

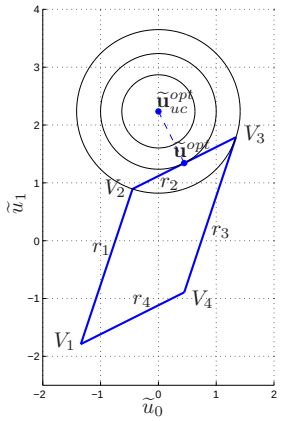


***Figura 2.2:*** Problema de optimización tras cambio de variable.

*En la figura se muestra también cómo obtener el óptimo* $\widetilde{\mathbf{u}}^{opt}$*, proyectando* $\widetilde{\mathbf{u}}_{uc}^{opt}$ *sobre la recta* $r_2$*:*

$$\widetilde{\mathbf{u}}^{opt} = \begin{bmatrix} 0{,}4472 \\ 1{,}3416 \end{bmatrix} \tag{2.16}$$

*Una vez se tiene el óptimo en las nuevas coordenadas, es directo obtener el óptimo para el problema de optimización original:*

$$\mathbf{u}^{opt} = H^{-1/2}\widetilde{\mathbf{u}}^{opt} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \tag{2.17}$$

■

### 2.3.3. Caracterización explícita del control predictivo

En este apartado, se pretende obtener una caracterización explícita del control predictivo haciendo uso de los argumentos geométricos de la programación cuadrática que se acaban de describir. Por tanto, el objetivo es resolver el problema de programación cuadrática (2.13) que surge al plantear un control predictivo y aplicar el cambio de variable (2.12).

Para ello, se hará uso del hecho de que, para el problema descrito, siempre es cierta una de las dos siguientes alternativas:

- El óptimo $\widetilde{\mathbf{u}}_{uc}^{opt}$ satisface todas las restricciones (2.14). En este caso, es evidente que $\widetilde{\mathbf{u}}^{opt} = \widetilde{\mathbf{u}}_{uc}^{opt}$.
- El óptimo $\widetilde{\mathbf{u}}_{uc}^{opt}$ no satisface todas las restricciones (2.14). En este caso, el óptimo $\widetilde{\mathbf{u}}^{opt}$ será un punto en la frontera del espacio definido por dichas restricciones. Es decir, para el valor del óptimo algunas de las restricciones no serán desigualdades estrictas, sino que serán igualdades. Se dice que estas restricciones se encuentran *activas*.

En ambos casos se pretende obtener la caracterización explícita, entendiendo como tal dos cosas: el óptimo solución del problema expresado como una función explícita de $x$, y la región del espacio $x$ en la que es válida dicha solución. Analizando el primer caso, es evidente que la caracterización puede obtenerse de manera directa, puesto que el óptimo sin restricciones viene expresado en función de $x$ y la región en la que es válido dicho óptimo puede obtenerse a partir de las ecuaciones de las restricciones (2.14):

$$\begin{cases} \widetilde{\mathbf{u}}^{opt}(x) = -H^{-1/2}Fx \\ -\widetilde{\Phi}H^{-1/2}Fx \leq \Delta - \Lambda x \end{cases}$$

Queda por tanto analizar el segundo caso, cuando $\widetilde{\mathbf{u}}_{uc}^{opt}$ no satisface (2.14). Como se ha visto, en este caso debe haber una o varias restricciones activas para la solución. Se define el *conjunto activo* $l := \{l_1, l_2, \ldots, l_{\widetilde{N}}\}$ como el conjunto que identifica los índices de las restricciones que se encuentran activas, donde $\widetilde{N}$ es el número de restricciones activas. Es decir, $l$ contiene las filas de la matriz $\widetilde{\Phi}$ que son igualdades, y no desigualdades estrictas. Se define $\widetilde{\Phi}_l$ como la submatriz de $\widetilde{\Phi}$ que sólo contiene esas filas. De manera complementaria, se define el *conjunto inactivo* $s = \{s_1, s_2, \ldots, s_{q-\widetilde{N}}\}$ como el conjunto de restricciones que no se encuentran activas. De la definición de ambos conjuntos, se deduce:

$$\widetilde{\Phi}_l \widetilde{\mathbf{u}}^{opt} = \Delta_l - \Lambda_l x \tag{2.18}$$

$$\widetilde{\Phi}_s \widetilde{\mathbf{u}}^{opt} < \Delta_s - \Lambda_s x \tag{2.19}$$

Tras el cambio de coordenadas (2.12), el óptimo puede obtenerse proyectando, en el espacio de las acciones de control, el óptimo sin restricciones sobre la

región definida por dichas restricciones. Es decir, el óptimo es aquel punto que, satisfaciendo las restricciones, está más cercano en distancia euclídea a la solución sin restricciones. Por tanto, nuestra solución vendrá dada al intersectar la cara activa (2.18) con el hiperplano perpendicular a ésta que pasa por $\widetilde{\mathbf{u}}_{uc}^{opt}$. Se puede demostrar [GS05] que dicha solución se puede expresar como:

$$\widetilde{\mathbf{u}}^{opt}(x) = \widetilde{\Phi}_l^{\dagger}(\Delta_l - \Lambda_l x) + [I - \widetilde{\Phi}_l^{\dagger}\widetilde{\Phi}_l]\widetilde{\mathbf{u}}_{uc}^{opt} \tag{2.20}$$

donde $\widetilde{\Phi}_l^{\dagger} = \widetilde{\Phi}_l^T[\widetilde{\Phi}_l\widetilde{\Phi}_l^T]^{-1}$ es la pseudoinversa de la matriz $\widetilde{\Phi}_l$. Dicha expresión es válida únicamente si el número de filas de $\widetilde{\Phi}_l$ es menor o igual al número de columnas. No obstante, esta condición se cumplirá siempre puesto que no es posible tener más restricciones activas que variables tiene el problema de optimización.
Sustituyendo la expresión del óptimo sin restricciones, se obtiene una expresión explícita del valor del óptimo:

$$\widetilde{\mathbf{u}}^{opt}(x) = \widetilde{\Phi}_l^{\dagger}(\Delta_l - \Lambda_l x) - [I - \widetilde{\Phi}_l^{\dagger}\widetilde{\Phi}_l]H^{-1/2}Fx \tag{2.21}$$

Para obtener el óptimo en las coordenadas originales, basta con deshacer el cambio (2.12):

$$\mathbf{u}^{opt}(x) = H^{-1/2}\widetilde{\mathbf{u}}^{opt}(x) \tag{2.22}$$

A continuación, sólo queda saber para qué subconjunto del espacio de estados es válida la solución anterior. Para ello, se introduce el concepto de *region activa* definido como el conjunto de puntos para los que el punto más cercano a $\widetilde{\mathbf{u}}_{uc}^{opt}$ y que satisface todas las restricciones (2.14) se encuentra sobre la cara activa (2.18). Por tanto, para cualquier $x$ que se encuentre en dicha región, la expresión del óptimo es (2.20).

Geométricamente, la región activa viene delimitada por [GS05]:

- Los hiperplanos que contengan a la cara activa correspondiente (2.18) y sean normales a alguna de las caras que comparten con la cara activa todas las restricciones activas excepto una. La expresión que define estos hiperplanos es:

$$[\widetilde{\Phi}_l\widetilde{\Phi}_l^T]^{-1}(\Lambda_l - \widetilde{\Phi}_l H^{-1/2}F)x \leq [\widetilde{\Phi}_l\widetilde{\Phi}_l^T]^{-1}\Delta_l \tag{2.23}$$

- Los hiperplanos normales a la cara activa (2.18) y que contengan una de las caras que, además de compartir con la cara activa todas las restricciones activas, tienen una restricción activa más. Estos hiperplanos quedan definidos por:

$$\left(-\widetilde{\Phi}_s\widetilde{\Phi}^{\dagger}(\Lambda_l - \widetilde{\Phi}_l H^{-1/2}F) + (\Lambda_s - \widetilde{\Phi}_s H^{-1/2}F)\right)x \leq (\Delta_s + \widetilde{\Phi}_s\widetilde{\Phi}_l^{\dagger}\Delta_l) \tag{2.24}$$

Nótese que, tal y como se ha definido el concepto de región activa, ésta está definida en el espacio de las acciones de control, mientras que las expresiones

(2.23) y (2.24), tal y como se desea para una solución explícita del problema, están definidas en el espacio de estados. La traslación de uno a otro es sencilla teniendo en cuenta que se conoce la relación entre el óptimo sin restricciones y el valor de las variables de estado: $\widetilde{\mathbf{u}}_{uc}^{opt}(x) = -H^{-1}Fx$. Por tanto, cualquier región definida como $R\widetilde{\mathbf{u}}_{uc}^{opt} \leq r$ puede escribirse como $-RH^{-1}Fx \leq r$.

No obstante, se observa que en general las dimensiones de $x$ y $\widetilde{\mathbf{u}}_{uc}^{opt}$ no tienen por qué coincidir. Si $n < Mm$, el espacio de estados es de dimension menor que el de las acciones de control $\mathbb{R}^{Mm}$ y por tanto, algunas de las regiones en la partición del espacio de estados obtenidas mediante las ecuaciones (2.23) y (2.24) pueden estar vacías. En este caso, será necesario comprobar si efectivamente hay regiones vacías y eliminar restricciones redundantes. Por el contrario, si $n \geq Mm$ las regiones que se obtendrán con las expresiones anteriores darán directamente la partición en el espacio de estados, por lo que no será necesario ningún postproceso.

**Ejemplo 2.3.** *Continuando con el problema planteado en los ejemplos 2.1 y 2.2,se pretende obtener la caracterización explícita de una de sus regiones. Para ello, se toma una cara activa, por ejemplo:*

$$\widetilde{\Phi}_l = [-0{,}447 \quad 0{,}894], \quad \Delta_l = 1, \quad \Lambda_l = 0$$

*Esta restricción se corresponde con $u_1 = 1$ antes de hacer el cambio de variable y está representada por la recta $r_2$ en la figura 2.2.*

*Para obtener la solución explícita, puede aplicarse la expresión* (2.20)*:*

$$\widetilde{\mathbf{u}}^{opt}(x) = \begin{bmatrix} -0{,}447 \\ 0{,}894 \end{bmatrix} + \begin{bmatrix} 0{,}894 & 0 \\ 0{,}447 & 0 \end{bmatrix} x$$

*Deshaciendo el cambio de variable:*

$$\mathbf{u}^{opt}(x) = H^{-1/2}\widetilde{\mathbf{u}}^{opt}(x) = \begin{bmatrix} -1 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} x$$

*Como era de esperar, la solución queda sobre la restricción activa, y por lo tanto $u_1^{opt} = 1$ sin depender del valor de $x$.*
*Por otro lado, como se ha visto, dicha solución será válida en una región que viene definida por las desigualdades* (2.23) *y* (2.24)*:*

$$\begin{bmatrix} & & [-1 & 3]x \leq 1 \\ \begin{bmatrix} 1 & 0 \\ -1 & 0 \\ 0 & 0 \end{bmatrix} x \leq & & \begin{bmatrix} 0 \\ 2 \\ 0 \end{bmatrix} \end{bmatrix}$$

*La primera desigualdad viene marcada por aquellos hiperplanos que comparten con la cara activa todas las restricciones activas excepto una. No obstante, cómo dicha cara sólo tiene una restricción activa, el único hiperplano es el correspondiente a la propia cara activa.*
*En cuanto al resto de desigualdades, hay que buscar las caras que tienen una restricción activa además de la cara activa. En este caso, se trata de $V_2$ y*

*$V_3$ en la figura 2.2, ya que para ambos vértices está activa la restricción que define $r_2$ además de la que define $r_1$ y $r_3$ respectivamente. Como se ha visto, las desigualdades que definen la región activa son los hiperplanos (en este caso rectas) perpendiculares a la cara activa y que contienen dichos vértices. Del resultado de* (2.24) *puede deducirse también que la última fila no da ninguna desigualdad válida. Esto se explica porque las rectas $r_2$ y $r_4$ no se cortan, es decir, no hay ningún punto que satisfaga simultáneamente la restricciones que definen estas rectas.*
*Se ha mostrado cómo obtener la caracterización de una de las regiones activas. Cubriendo todas las posibles combinaciones de restricciones activas, se completa la caracterización del control predictivo para todo el espacio de estados. Así, se tienen 9 regiones diferentes, tal y como se muestra en la figura 2.3, dentro de cada una de las cuales, el óptimo se calcula proyectando sobre un elemento diferente, de forma que,en función de donde se encuentre $\tilde{\mathbf{u}}_{uc}^{opt}$, $\tilde{\mathbf{u}}^{opt}$ estará sobre una de las rectas $r_i$ o en uno de los vértices $V_i$.* ■

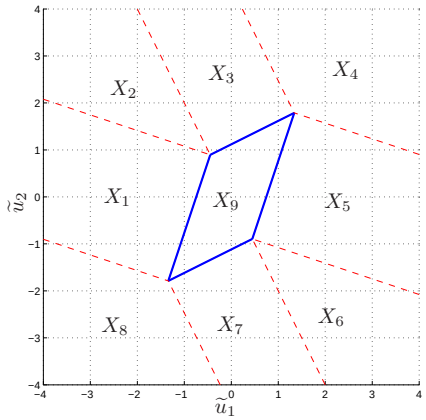


***Figura 2.3:*** Regiones del controlador predictivo explícito.

Una vez visto cómo resolver el problema de optimización, queda introducir la filosofía del horizonte móvil, que implica aplicar la solución óptima únicamente en el primer instante. Esto se realiza aplicando únicamente los primeros $m$ elementos de $\mathbf{u}^{opt}(x)$:

$$\mathcal{K}_{N,M}(x) = u_0^{\text{opt}}(x) = [I \quad 0 \dots 0]\mathbf{u}^{opt}(x) \tag{2.25}$$

Combinando (2.25) con (2.21) y (2.22) podemos expresar la ley de control:

$$\mathcal{K}_{N,M}(x) = G_i x + h_i, \quad \text{para } x \in X_i, \quad i = 0, \dots, N_r \tag{2.26}$$

donde $N_r$ es el número de regiones $X_i$ en que tenemos una solución explícita diferente y las matrices $G_i$ y $h_i$ se obtienen operando a partir de las expresiones anteriormente citadas.

Las regiones $X_i$ en las que son válidas las anteriores expresiones explícitas, vienen dadas por las ecuaciones (2.23) y (2.24). Para una mayor simplicidad,

se agrupan ambos conjuntos de ecuaciones en una notación más compacta:

$$X_i := \{x \in \mathbb{R}^n \,|L_i x \leq W_i\} \tag{2.27}$$

Se puede demostrar que todas las regiones así definidas forman una partición:

$$\mathbb{X} := \{X_i \,|i = 1, \ldots, N_r\} \tag{2.28}$$

y por tanto cumplen:

$$\text{int}(X_i) \cap \text{int}(X_j) = \emptyset, \quad i \neq j$$

donde int hace referencia al interior de la región.

### 2.3.4. Caracterización explícita mediante las condiciones KKT

Se supone un problema de optimización estándar:

$$\begin{array}{lll} \min & f_0(x) & \\ \text{sujeto a:} & f_i(x) \leq 0 & i = 1, \ldots, m \\ & h_i(x) = 0 & i = 1, \ldots, p \end{array} \tag{2.29}$$

A partir del problema anterior, puede formularse el problema dual como:

$$\begin{array}{ll} \max & g(\lambda, \nu) \\ \text{sujeto a:} & \lambda \geq 0 \\ & g(\lambda, \nu) = \inf \left( f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} \nu_i h_i(x) \right) \end{array} \tag{2.30}$$

La función a maximizar en (2.30), $g(\lambda, \nu)$, es la llamada *función dual de Lagrange*, y es sencillo demostrar que, para cualquier valor de $\lambda \geq 0$ y $\nu$, representa una cota inferior del óptimo solución del problema primal [BV04]. El problema dual, por tanto, busca la máxima cota inferior del mínimo solución del problema (2.29), que puede ser una buena aproximación del propio mínimo. De hecho, bajo ciertas condiciones, ambos valores coinciden, en cuyo caso se dice que ambos son óptimos duales fuertes. Cuando esto no sucede, se dice que existe dualidad débil entre ambos problemas.

Si se tiene cualquier pareja de óptimos duales fuertes $x^*$ y $(\lambda^*, \nu^*)$ (soluciones de (2.29) y (2.30) respectivamente) y se asume que las funciones $f_i$ y $h_i$ son diferenciables, se puede comprobar que se satisface:

$$\begin{array}{rl} f_i(x^*) \leq 0, & i = 1, \ldots, m \\ h_i(x^*) = 0, & i = 1, \ldots, p \\ \lambda_i^* \geq 0, & i = 1, \ldots, m \\ \lambda_i^* f_i(x^*) = 0, & i = 1, \ldots, m \\ \nabla f_0(x^*) + \sum_{i=1}^{m} \lambda_i^* \nabla f_i(x^*) + \sum_{i=1}^{p} \nu_i^* \nabla h_i(x^*) = 0, & \end{array}$$

que se llaman condiciones de *Karush-Kuhn-Tucker* (condiciones KKT).
Se trata por tanto de condiciones necesarias para que el conjunto $(x^*, \lambda^*, \nu^*)$ sea solución de los problemas (2.29) y (2.30).
Si además el problema primal es convexo, las condiciones anteriores no son sólo necesarias, sino también suficientes, y por lo tanto pueden ser utilizadas para calcular dicho óptimo.

Volviendo al problema (2.9), por simplicidad se realiza el cambio de variable $z = \mathbf{u} + H^{-1}Fx$, con lo que el problema de optimización queda:

$$\begin{aligned} \min \quad & \tfrac{1}{2} z^T H z, \\ \text{sujeto a:} \quad & \Phi z \leq \Delta + Sx \end{aligned}$$

donde $S = -\Lambda + \Phi H^{-1} F$.

Recordando que $H \succ 0$, se pueden escribir las condiciones KKT, que serán necesarias y suficientes [BMDP02]:

$$\Phi z - \Delta - Sx \leq 0 \tag{2.31a}$$
$$\lambda \geq 0 \tag{2.31b}$$
$$\lambda_i(\Phi^i z - \Delta^i - S^i x) = 0 \tag{2.31c}$$
$$Hz + \Phi^T \lambda = 0, \tag{2.31d}$$

La idea desarrollada en [BMDP02], que tiene paralelismos con el enfoque geométrico visto en la sección anterior, se basa en el hecho de que, si se conoce el conjunto de restricciones que se encuentran activas, $l$, es posible obtener la solución al problema de optimización a partir de las ecuaciones (2.31c) y (2.31d) como:

$$z = H^{-1}\Phi_l^T \left(\Phi_l H^{-1} \Phi_l^T\right)^{-1} (\Delta_l + S_l x)$$

Puede comprobarse que, deshaciendo los cambios de variable realizados en los dos enfoques diferentes planteados, la expresión anterior coincide con la obtenida de (2.21) y (2.22).
En cuanto a la caracterización de la región para la que es válida la solución anterior, basta con reemplazar el óptimo obtenido en (2.31a) y (2.31b):

$$\begin{aligned} \Phi H^{-1}\Phi_l^T \left(\Phi_l H^{-1} \Phi_l^T\right)^{-1} (\Delta_l + S_l x) &\leq \Delta + Sx \\ -\left(\Phi_l H^{-1} \Phi_l^T\right)^{-1} (\Delta_l + S_l x) &\geq 0 \end{aligned}$$

En este caso, puede comprobarse que las ecuaciones obtenidas coinciden con (2.23) y (2.24).

### 2.3.5. Obtención de todas las regiones de la solución explícita

Hasta el momento, en los procedimientos para obtener la caracterización del control predictivo se ha partido siempre del conocimiento de qué restricciones se encuentran activas. Para completar la metodología, es necesaria una manera de

enumerar todas las posibles combinaciones de restricciones activas para nuestro problema. A continuación, se describen diferentes enfoques para resolver este problema existentes en la bibliografía.

### Búsqueda exhaustiva

En [SGD03] los autores proponen considerar todas las posibles combinaciones de restricciones activas y obtener la caracterización de la región para cada una de ellas, descartando posteriormente aquellas regiones que estén vacías. El número de regiones iniciales a considerar es de $N_r = 2^q$, donde $q = Mm$ es el máximo número de restricciones que puede haber simultáneamente activas, es decir, el número de grados de libertad del problema de optimización. Evidentemente, en la gran mayoría de los casos muchas de esas $N_r$ regiones estarán vacías, por lo que éste es un método bastante costoso computacionalmente.

### Exploración y partición del espacio de estados

Este método, propuesto en [BMDP02] para resolver problemas de programación cuadrática multiparamétrica ($mpQP$), se basa en buscar un punto factible del espacio de restricciones (por ejemplo resolviendo un problema LP) y, para dicho punto, resolver el problema QP. Con la solución obtenida es posible comprobar qué restricciones se encuentran activas y, por lo tanto, caracterizar la región que le corresponde. Posteriormente, aplicando el algoritmo propuesto en [DP00], se van invirtiendo cada una de las restricciones que definen la región activa. Esto proporciona una nueva región para la que se vuelve a buscar un punto factible y a caracterizar la región activa que le corresponde (por ejemplo mediante las condiciones KKT). De manera recursiva, se obtienen todas las regiones críticas. Un inconveniente que presenta este método es que una región crítica con una determinada solución afín, puede ser dividida innecesariamente, por lo que será necesario posteriormente aplicar algoritmos de unión de regiones como los de [BFT01].
En [BBM03b], los autores se basan también en el algoritmo de [DP00], pero en lugar de usarlo para intersectar con las condiciones de optimalidad de KKT y caracterizar las regiones, se usa simplemente para guiar la exploración del espacio. En este caso, algunas regiones pueden aparecer duplicadas, y será necesario eliminarlas, comprobando la coincidencia de todas las restricciones activas.

### Desplazamiento sobre la frontera de la región

Esta idea, computacionalmente más eficiente que las anteriores, es utilizada en [Bao] y [GBTM04]. Se basa en, dada una región crítica conocida, tomar un punto de una de las facetas que la definen (una de las desigualdades no redundantes que la describen) y desplazarlo una pequeña distancia en dirección perpendicular. Esto nos dará un punto que se encuentra en una región diferente a la anterior. De nuevo, resolviendo el QP es posible averiguar qué restricciones

se encuentran activas y por tanto caracterizar esta nueva región. Siguiendo este procedimiento para todas las facetas de todas las regiones, se puede caracterizar la partición completa del espacio de estados.

### Algoritmo para mpQP de Tøndel et al.

En [TJB03a] se introduce un algoritmo todavía más eficiente, basado en que las restricciones activas en la región de interés se pueden determinar a partir de las restricciones activas de una región adyacente examinando el hiperplano que las separa. Los autores proponen una metodología que, dada una región (definida por un conjunto de restricciones activas $l$) y una faceta ($f$), es válida si se cumplen las siguientes condiciones:

- Para la región activa definida por el conjunto $l$, las filas de la matriz $\Phi_l$ son linealmente independientes.
- En la definición de la región (antes de eliminar las redundancias) no hay más de una desigualdad que defina $f$.
- Para cualquier punto de la región, el óptimo nunca presenta restricciones activas débilmente (es decir, ningún óptimo satisface $\lambda_i = 0$ para $i \in l$).

Si dichas condiciones se cumplen, se comprueba qué tipo de desigualdad corresponde a la faceta. Si viene de una condición del tipo (2.31a), el óptimo en la región actual no tiene activa la restricción correspondiente. No obstante, al pasar al otro lado de la faceta, dicha restricción pasará a estar activa. Si por contra la faceta viene dada por una restricción del tipo (2.31b), en la región actual la restricción estará activa y, por lo tanto, al cruzar la faceta dejará de estarlo.
El algoritmo por tanto consistirá en, dado un punto factible conocido, obtener la caracterización de su región crítica correspondiente y retener la información del tipo de hiperplanos que la definen. A continuación, para cada faceta se obtiene la región contigua correspondiente, de la que se conocen directamente las restricciones activas, y por tanto su caracterización completa. Recorriendo iterativamente las facetas de todas las regiones, se completa la descripción de toda la partición del estado.
Las ventajas de este algoritmo frente a los anteriores descritos son el ahorro de la resolución de problemas QP (para comprobar las condiciones KKT) y evitar dividir regiones innecesariamente.
Los autores proponen también algoritmos para los casos degenerados en que las condiciones impuestas no se cumplen. No obstante, como se verá a continuación, dichos algoritmos realizan implícitamente una suposición que, en el caso general, no tiene por qué cumplirse.

### Propiedad de faceta-a-faceta y modificación de algoritmos previos

En [SKJ+06], los autores muestran como, implícitamente, los algoritmos para enumerar todas las regiones existentes presentados en [Bao],[GBTM04] y [TJB03a] consisten en una misma estrategia de exploración del espacio de estados (resumida en el algoritmo 2.1) y diferentes métodos para obtener las regiones adyacentes a una conocida.

**Algoritmo 2.1** Exploración del espacio de estados

1. Buscar un punto $x$ factible y caracterizar su región crítica, $X_1$.
2. Inicializar conjunto de regiones conocidas $\mathbb{R} := \{X_1\}$ y de regiones inexploradas $\mathbb{U} := \{X_1\}$.
3. Mientras $\mathbb{U} \neq \emptyset$:
   *a*) Elegir un elemento $U \in \mathbb{U}$ y hacer $\mathbb{U} = \mathbb{U} \backslash U$.
   *b*) Para todas las facetas $f$ de $U$:
   - Buscar las regiones $S_i$ adyacentes a $U$ a través de $f$.
   - $\mathbb{U} = \mathbb{U} \cup \{S_i \backslash \mathbb{R}\}$
   - $\mathbb{R} = \mathbb{R} \cup S_i$

Además, todos ellos se basan en lo que se define como *propiedad de faceta-a-faceta*:

**Definición 2.1.** Dada una partición poliédrica de la forma (2.28) con poliedros de dimensión $\mathbb{R}^n$, se dice que la propiedad de faceta-a-faceta se cumple si $f_{ij} := X_i \cap X_j$ es una faceta tanto de $X_i$ como de $X_j$ para cualquier pareja $(X_i, X_j) \in \mathbb{X}$ cuya intersección tenga dimensión $n-1$. □

La definición anterior implica que, para una partición que cumpla la propiedad faceta-a-faceta, una región no puede tener más de una región adyacente por cada una de sus facetas. Es evidente que los algoritmos descritos en los trabajos enumerados anteriormente hacen uso implícito de esta propiedad, puesto que, cuando se conoce la caracterización de una región, para encontrar una nueva simplemente se invierte el sentido de una de sus facetas (suponiendo que de esta forma va a encontrarse una única región). No obstante, en [SKJ+06] se muestra como, en general, no puede garantizarse que la propiedad de faceta-a-faceta se cumpla, por lo que los algoritmos anteriores serían incorrectos.

Por otro lado, puede demostrarse que cuando se cumplen las condiciones de [TJB03a] enumeradas anteriormente, la propiedad de faceta-a-faceta sí se cumple. Por ello, y dado que ése es el algoritmo más eficiente, los autores de [SKJ+06] proponen utilizarlo cuando las condiciones se cumplan y, en el resto de casos, utilizar el método de exploración y partición de [BMDP02], que no se apoya en la propiedad de faceta-a-faceta. El algoritmo 2.2 resume este

procedimiento para obtener las regiones adyacentes a una dada a través de una determinada faceta $f$.

**Algoritmo 2.2** Búsqueda de regiones adyacentes a la región $X$ a través de la faceta $f$

$$X = \{x \,|L_i x \leq W_i, i = 1, \ldots, q\}$$

1. Si se cumplen las condiciones para el algoritmo de [TJB03a], aplicarlo y obtener la única región adyacente a $X$ a través de $f$.
2. Si no se cumplen, tomar un escalar $\varepsilon > 0$ y formar el poliedro:

$$M := \left\{ x \;\middle|\; \begin{array}{cc} L_i x \leq W_i, & i \in \{1, \ldots, q\} \backslash \{j\} \\ L_j x \geq W_j & \\ L_j x \leq W_j + \varepsilon & \end{array} \right\}$$

   siendo $j$ la desigualdad que describe la faceta $f$.
3. Buscar las regiones de la partición $\mathbb{X}$ que tienen intersección de dimensión completa con $M$ usando el algoritmo de [BMDP02].
4. Para cada una de ellas, comprobar si tiene intersección de dimensión $n-1$ con $X$: si la tiene es una región adyacente.

### 2.3.6. Algoritmo en línea

Como se ha visto en las secciones anteriores, el problema de optimización presente en el control predictivo puede resolverse de manera explícita con la ley de control afín a tramos (2.26), cuyas regiones son lineales y están descritas de la forma (2.27). Por tanto, si se tienen almacenadas en una tabla las diferentes regiones lineales junto con las soluciones afines que les corresponden, es posible calcular la acción de control necesaria en cada instante de muestreo identificando en cual de las regiones $X_i$ se encuentra el estado actual y aplicando la expresión afín correspondiente.

Aunque es cierto que de esta manera se evita tener que resolver en línea un problema de programación cuadrática, la solución explícita también tiene un coste computacional no despreciable, principalmente asociado a decidir en cuál de las regiones lineales se encuentra el estado del sistema. Por tanto, es importante resolver dicho subproblema de la manera más eficiente posible. A continuación se describen algunas de las propuestas existentes para ello.

#### Búsqueda secuencial

Dado un valor $x$ de las variables de estado, la manera más directa de determinar en qué región se encuentra es un algoritmo de búsqueda secuencial que

vaya comprobando, para cada región $X_i$, si se cumple $L_i x \leq W_i$. Cuando se encuentra la región para la que dicha desigualdad se cumple, se aplica la función afín correspondiente que define la acción de control a aplicar: $u(x) = G_i x + h_i$. Es evidente que, en el peor de los casos, el algoritmo de búsqueda secuencial debe comprobar todas las regiones (con todos los hiperplanos que la definen) por lo que, para un total de $L$ hiperplanos, el coste computacional será:

$$C = 2nL$$

correspondiente a realizar $n$ productos y $n$ sumas para cada hiperplano.
Por tanto, para problemas grandes el coste computacional será también elevado, puede que por encima de la resolución de un QP (teniendo en cuenta que éste es un problema de optimización muy generalizado para el que existen algoritmos eficientes).

### Función descriptora afín a tramos

En [BBBM01] se proponen algoritmos que, basados en las propiedades de la función de coste y la ley de control óptima del problema de optimización, mejoran tanto la eficiencia del cálculo en línea de la solución explícita como los requerimientos de memoria.
El algoritmo consiste en buscar una función descriptora continua y afín a tramos, de la forma:

$$f(x) = \{f_i(x) = A_i x + B_i \,|x \in X_i, i = 1, \dots N_r\}$$

con

$$A_i \neq A_j, \quad \forall j \in C_i, \quad i = 1, \dots, N_r$$

donde $C_i$ contiene la lista de todos los polinomios de la partición vecinos de $X_i$. A partir de dicha función, se define una lista $O_i(x) = \{o_j^i(x) \,|j \in C_i\}$ donde:

$$\begin{cases} +1 & f_i(x) \geq f_j(x) \\ -1 & f_i(x) < f_j(x) \end{cases}$$

Los autores demuestran que la lista $O_i(x)$ toma un valor constante, $S_i$, para cualquier $x \in X_i$. Sin embargo, para un $x \notin X_i$, dicha lista es diferente a $S_i$. Esto permite diseñar un algoritmo en el que, en lugar de tener almacenadas todas las regiones polinomiales, simplemente se calculan fuera de línea y almacenan las listas $C_i$ y $S_i$. El algoritmo 2.3 describe el procedimiento seguido en el algoritmo en línea.
El algoritmo 2.3 tiene un coste, en el peor de los casos de:

$$C = (2n - 1)N_r + L$$

correspondiente a $N_r n$ multiplicaciones, $N_r(n-1)$ sumas y $L$ comparaciones. Dado que, en general, $N_r \ll L$, el coste de este algoritmo es bastante menor que el de búsqueda secuencial.

En cuanto a la obtención de las funciones descriptoras, los autores realizan dos propuestas:

**Algoritmo 2.3** Algoritmo en línea mediante funciones descriptoras

1. partir de la región $i = 0$.
2. Calcular $O_i(x)$
3. Comprobar si $O_i(x) = S_i$
   *a*) Si son iguales, $x \in X_i$,
   *b*) Si no, volver al paso 2 con $i = C_i(q)$, donde $o_q^i(x) \neq s_q^i$

- Generar la función a partir del gradiente de la función de coste. Consiste en buscar fuera de línea un vector $w$ que no sea paralelo a ninguno de los hiperplanos de la partición poliédrica y definir las funciones como:

$$f_i(x) = A_i x + B_i, \quad A_i = 2w^T Q_i, \quad B_i = w^T T_i$$

  donde $Q_i$ y $T_i$ son las matrices y vectores, respectivamente, que definen cada uno de los tramos de la función de coste:

$$V(x) = \left\{ \; q_i(x) = x^T Q_i x + T_i^T x + V_i, \quad \text{para } x \in X_i, \quad i = 1, \ldots, N_r \; \right\}$$

- Generar la función a partir de la solución explícita del problema de optimización:

$$f(x) = w' U^{opt}(x)$$

  donde $w$ se define como en el caso anterior y

$$U^{opt}(x) = \left\{ \; u_i(x) = G_i x + h_i, \quad \text{para } x \in X_i, \quad i = 1, \ldots, N_r \; \right\}$$

### Árbol de búsqueda binario

Una posibilidad para reducir todavía más el coste computacional de la búsqueda secuencial es la propuesta en [TJB03b], basada en explotar la convexidad de los conjuntos poliédricos para construir un árbol binario de búsqueda en el que, en cada nivel (o nodo), se evalúe una única desigualdad lineal (hiperplano) y se compruebe su signo. Dependiendo de dicho signo, se avanza hacia el subnodo izquierdo o derecho. Un vez recorrido el árbol, se llega a un nodo hoja que tiene asociada una ley de control afín única. El algoritmo 2.4 resume dicho procedimiento.

En cuanto al algoritmo fuera de línea que permita obtener el árbol binario, el objetivo fundamental es el de minimizar la profundidad del árbol ($D$) puesto que éste parámetro afecta al coste computacional del algoritmo 2.4:

$$C = (2n + 1)D + 2nm \tag{2.32}$$

donde $n$ es el número de estados y $m$ el número de entradas.
Como segundo criterio, se intenta hacer mínimo el número de nodos en el

**Algoritmo 2.4** Recorrido del árbol binario

1. Asignar al nodo actual, $N_k$, el nodo raíz.
2. Mientras $N_k$ no sea un nodo hoja:
   a) Evaluar el hiperplano $d(x) = a^T x - b$ correspondiente a $N_k$.
   b) Dependiendo del signo de $d(x)$ asignar a $N_k$ el hijo del nodo actual correspondiente.
3. Evaluar la ley de control $u(x)$ correspondiente a $N_k$.

árbol, puesto que éste afecta a la memoria necesaria para almacenar las tablas correspondientes.
Para la obtención del árbol binario, son necesarias una serie de definiciones. Dado una representación afín de un controlador predictivo como la dada por (2.26) y (2.27), consideramos el conjunto de todos los hiperplanos que definen las diferentes regiones $X_i$:

$$d_j(x) = L_j x - W_j \quad \text{para } j = 1, \ldots, L \tag{2.33}$$

y definimos:

- $\mathcal{J}$: La representación de índices de un poliedro como una combinación de índices y signos de $d_j$. Por ejemplo, $\mathcal{J} = \{1^+, 2^-, 5^+\}$ representaría al poliedro definido por $d_1 \geq 0$, $d_2 \leq 0$ y $d_5 \geq 0$.
- $\mathcal{I}(\mathcal{J})$: El conjunto de regiones poliédricas correspondientes a $\mathcal{J}$. Se trata de todas las regiones $X_i$ que satisfacen todas las restricciones definidas en $\mathcal{J}$.
- $\mathcal{F}(\mathcal{I})$: El conjunto de funciones afines correspondientes a un conjunto $\mathcal{I}$; $\mathcal{F}(\mathcal{I}) = \{k \,|\mathcal{K}_k$ corresponde a $X_i, i \in \mathcal{I}\}$.

La idea fundamental para la construcción de un árbol de profundidad mínima es la siguiente: se pretende que el hiperplano que se asigna a cada nodo $N_k$ divida la parte del árbol que queda por debajo en dos partes lo más iguales posibles. Es decir, idealmente se pretende elegir un hiperplano $j_k$ tal que el número de funciones afines válidas para regiones que satisfacen todas las restricciones de nodos anteriores, $|\mathcal{F}(\mathcal{I}(\mathcal{J}_k))|$, quede dividido en dos subconjuntos con el mismo número de regiones: $|\mathcal{F}(\mathcal{I}(\mathcal{J}_k \cup j_k^+))| = |\mathcal{F}(\mathcal{I}(\mathcal{J}_k \cup j_k^-))| = \frac{1}{2}|\mathcal{F}(\mathcal{I}(\mathcal{J}_k))|$. Como en un caso general no es posible dividir el conjunto $|\mathcal{F}(\mathcal{I}(\mathcal{J}_k))|$ en dos subconjuntos con exactamente el mismo número de elementos, se intenta que la diferencia sea lo menor posible: $j_k = \arg\min_j \max(|\mathcal{F}(\mathcal{I}(\mathcal{J}_k \cup j_k^+))|, |\mathcal{F}(\mathcal{I}(\mathcal{J}_k \cup j_k^-))|)$.
Nótese en este punto que $\mathcal{I}(\mathcal{J}_k \cup j_k^{\pm}) \subseteq (\mathcal{I}(\mathcal{J}_k) \cap \mathcal{I}(j\pm))$, es decir, las regiones $X_i$ que satisfacen simultáneamente todas las restricciones en $\mathcal{J}_k$ además de $j_k^{\pm}$, son un subconjunto de las regiones que satisfacen por un lado $\mathcal{J}_k$ y por el otro

$j_k^{\pm}$. Las regiones $X_i$ que pertenecen al segundo conjunto pero no al primero siempre están divididas por la restricción $j_k$ (véase lema 1 en [TJB03b]).
Por tanto, una forma de obtener el conjunto $\mathcal{I}(\mathcal{J}_k \cup j_k^{\pm})$ es obtener inicialmente los conjuntos $\mathcal{I}(\mathcal{J}_k)$, $\mathcal{I}(j_k^{+}))$ e $\mathcal{I}(j_k^{-}))$ y, para aquellas regiones $X_i$ que se encuentren en todos ellos, resolver dos LPs:

$$\begin{array}{ll} \min & \pm\, d_j(x) \\ \text{sujeto a:} & \begin{array}{l} L_i x \leq W_i \\ L_j x \leq W_j, \quad j \in \mathcal{J}_k \end{array} \end{array} \tag{2.34}$$

Las soluciones de estos LPs indicarán a qué lado de $d_j$ se encuentra el subconjunto de $X_i$ que satisface todas las restricciones de $\mathcal{J}_k$, o si la región es atravesada por dicho hiperplano. Nótese además que $(\mathcal{I}(\mathcal{J}_k) \cap \mathcal{I}(j^{\pm}))$ es sencillo de calcular, y puede utilizarse como aproximación de $\mathcal{I}(\mathcal{J}_k \cup j_k^{\pm})$.

El algoritmo 2.5 detalla el procedimiento descrito para la construcción del árbol binario. Los pasos más costosos son el 1, en el que hay que comprobar a que lado de cada uno de los $L$ hiperplanos se encuentra cada una de las $n_r$ regiones (resolviendo por tanto $2Ln_r$ LPs), y el 6, en el que los primeros $n_j$ conjuntos $\mathcal{I}(\mathcal{J}_k \cup j_k^{\pm})$ aproximados como $(\mathcal{I}(\mathcal{J}_k) \cap \mathcal{I}(j\pm))$ se calculan exactamente (resolviendo los correspondientes LPs). $n_j$ es un parámetro de diseño, que muestra el compromiso entre el coste del algoritmo de construcción del árbol binario (fuera de línea) y la profundidad de dicho árbol (y por tanto el coste del algoritmo en línea calculado (2.32)).

En cuanto a la complejidad $D$ de un determinado árbol binario, ésta en un caso ideal, en que en cada nodo se dividen las acciones de control en dos conjuntos exactamente iguales, sería de:

$$D = \lceil log_2(n_r) \rceil = \left\lceil \frac{ln(n_r)}{ln(2)} \right\rceil$$

Como realmente un árbol binario nunca va a poder construirse idealmente, se define un parámetro $\alpha$ de la siguiente forma:

$$\frac{\max(|\mathcal{F}_k^{-}|, |\mathcal{F}_k^{+}|)}{|\mathcal{F}_k|} \leq \alpha \text{ para todo } N_k$$

Es evidente que, en el caso ideal, $\alpha = 0{,}5$ y a medida que la división de los conjuntos es menos equilibrada, se acerca a $\alpha = 1$. Con este nuevo parámetro, se puede estimar la complejidad del árbol binario como:

$$D = \left\lceil -\frac{ln(n_r)}{ln(\alpha)} \right\rceil$$

Experimentalmente, los autores de [TJB03b] proponen como valor conservativo $\alpha = \frac{2}{3}$. En cualquier caso, independientemente del valor de $\alpha$, el coste computacional del algoritmo en línea 2.4 es logarítmico en el número de regiones, y por tanto en general más favorable que los del resto de propuestas.

**Algoritmo 2.5** Construcción del árbol binario

1. Calcular los conjuntos de índices $\mathcal{I}(j^+)$ e $\mathcal{I}(j^-)$ para cada $j = \{1, \ldots, L\}$.
2. Inicializar el nodo raíz del árbol como $N_1 \leftarrow (\{1, \ldots, n_r\}, \emptyset)$
3. Inicializar el conjunto de nodos inexplorados como $\mathcal{U} \leftarrow \{N_1\}$.
4. Seleccionar cualquier nodo inexplorado $N_k \in \mathcal{U}$ y hacer $\mathcal{U} \leftarrow \mathcal{U} \setminus N_k$.
5. Calcular las aproximaciones $\mathcal{I}(\mathcal{J}_k) \cap \mathcal{I}(j^{\pm})$ para cada j, y ordenar los hiperplanos por la cantidad $max(|\mathcal{F}(\mathcal{I}(\mathcal{J}_k) \cap \mathcal{I}(j^+))|, |\mathcal{F}(\mathcal{I}(\mathcal{J}_k) \cap \mathcal{I}(j^-))|)$.
6. Calcular exactamente $\mathcal{I}_k^{\pm} = \mathcal{I}(\mathcal{J}_k \cap j^{\pm})$ para cada uno de los $n_j$ primeros elementos en la lista ordenada del paso anterior. Esto se hace resolviendo los LPs (2.34) para cada $i \in \mathcal{I}(\mathcal{J}_k) \cap \mathcal{I}(j^+) \cap \mathcal{I}(j^-)$. Seleccionar entre ellos $j_k = argmin_j max(|\mathcal{F}(\mathcal{I}_k^+)|, |\mathcal{F}(\mathcal{I}_k^-)|)$.
7. Completar el nodo como $N_k \leftarrow j_k$, y crear dos nodos hijos, $N^{\pm} \leftarrow (\mathcal{I}_k^{\pm}, \mathcal{J}_k \cup j^{\pm})$.
8. Si $|\mathcal{F}(\mathcal{I}^{\pm})| > 1$, añadir $N^{\pm}$ a $\mathcal{U}$. Si no, $N^{\pm}$ es un nodo hoja, hacer $N^{\pm} \leftarrow \mathcal{F}(\mathcal{I}^{\pm})$.
9. Si $\mathcal{U} \neq \emptyset$, volver al paso 4. Si no, finalizar.

## 2.4. Control predictivo no lineal

Aunque la mayoría de procesos reales que aparecen en problemas de la ingeniería de control son no lineales, es el control predictivo lineal el que está más ampliamente extendido. Esto se debe fundamentalmente a dos razones [CB04]: la sencillez de la optimización necesaria (que como se ha visto puede realizarse mediante la resolución de un QP y que puede incluso calcularse explícitamente) y la validez de los modelos lineales para muchos procesos no lineales para un rango de funcionamiento acotado.

No obstante, existen casos en los que los modelos lineales no son suficientes para predecir el comportamiento del proceso, bien porque las no linealidades son demasiado severas, o bien porque el rango de funcionamiento es demasiado amplio.

Para estos casos, se hace necesaria la utilización de modelos no lineales, para los que la filosofía del control predictivo, entendida como la utilización de un modelo del proceso, optimización de un índice de coste y aplicación de un horizonte móvil, es también aplicable.

El principal inconveniente del uso de estos modelos es que en el caso general, el problema de optimización que aparece (2.1) es un problema de programación no lineal (NLP) en contraposición al problema de optimización cuadrática (QP). Este tipo de problemas se resuelve mediante programación cuadrática

secuencial (SQP), una técnica iterativa que básicamente consiste en sustituir en cada iteración la función de coste y restricciones no lineales del problema original por, respectivamente, una función cuadrática y restricciones lineales. Esto proporciona un problema de programación cuadrática cuya solución da una dirección de búsqueda para la solución del problema original que se usa en la siguiente iteración.
Esta técnica presenta diversas dificultades [Bie00]:

- En el método original son necesarias las segundas derivadas de las funciones no lineales, que no siempre pueden ser calculadas.
- La convergencia del algoritmo al óptimo global no está garantizada.
- En cada período de muestreo es necesario resolver varios QPs, lo que implica un coste computacional elevado.
- Las soluciones intermedias que se obtienen en el procedimiento iterativo no tienen por qué cumplir las restricciones originales. Este problema, junto con el anterior, puede provocar que si se acaba el tiempo para dar una acción de control y el procedimiento iterativo no ha terminado, no se tenga ni siquiera una solución factible.

Aunque la programación no lineal es actualmente un campo de investigación activo y existen propuestas que persiguen resolver las dificultades enumeradas, éstas son todavía insalvables para algunas aplicaciones de control predictivo, por lo que en muchas ocasiones se evita resolver un problema tan general y se intentan aprovechar algunas de las particularidades de cada problema.

Algunas de estas posibles soluciones, que se resumen en [CB04], son:

**MPC lineal extendido:** Introducido originalmente para el DMC [HA91]. Se añade un término no lineal al modelo de predicción pero no dependiente de las acciones de control futuras, lo que permite que el problema a optimizar siga siendo resoluble mediante un QP.

**Modelos locales:** Se trata de linealizar sucesivamente el modelo no lineal del sistema alrededor de diferentes puntos de funcionamiento. Dentro de esta estrategia destacan [KCR99], que realiza las sucesivas linealizaciones alrededor de las trayectorias predichas en instantes anteriores, y [TI01], que ajusta un conjunto de controladores predictivos generalizados (GPCs) para los diferentes modelos y aplica una planificación de ganancia entre ellos.

**MPC no lineal subóptimo:** Fue propuesto por [SMR99]. La idea consiste, en lugar de buscar la secuencia de acciones de control que minimiza un índice de coste, en buscar una que cumpla restricciones y que haga decrecer dicho índice una cantidad suficiente. Después, si queda tiempo para el cálculo, buscar una reducción mayor del coste.

**Uso de horizontes bajos:** La idea es calcular exactamente sólo la primera acción de control de la secuencia (la única que se va a aplicar) y aproximar de alguna manera el resto (por ejemplo, linealizando el modelo del sistema). Al reducir las variables de decisión, el problema optimización se simplifica considerablemente. En [ZZ01] y [KCR98] se utilizan estrategias de este tipo.

**Descomposición de la secuencia de control:** La idea en este caso consiste en separar la secuencia de acciones de control futuras en dos términos, una secuencia de control base que viene dada y un incremento sobre esta libre. Después, se separa la respuesta debida a cada una de ellas, usando para la primera un modelo no lineal y para la segunda uno lineal. Esto permite obtener la secuencia de incrementos libre mediante la resolución de un QP. Debido al no cumplimiento del principio de superposición para sistemas no lineales, en general es necesario establecer un procedimiento iterativo en el que la suma de las dos secuencias se hace ahora igual a una nueva secuencia base y se buscan nuevos incrementos. Cuando éstos se hacen igual a cero, se aplica la acción de control [Key98].

**Linealización por realimentación:** Consiste en una cancelación de la no linealidad del sistema, lo que en general lleva a tener un índice de coste cuadrático pero restricciones no lineales, que se aproximan por restricciones lineales [BBKC99].

**MPC basado en modelos de Volterra:** Cuando el modelo no lineal del sistema es un modelo de Volterra de segundo orden puede establecerse un procedimiento iterativo pero considerablemente rápido [DPO02] que, mediante la resolución de varios QP, obtiene el óptimo. Un caso especial de modelos de Volterra, modelos de tipo Hammerstein y Wiener, permiten resolver la optimización mediante una cancelación de la no linealidad.

**Redes neuronales:** Al ser las redes neuronales aproximadores universales, pueden también ser utilizadas como optimizador de la función de coste fuera de linea [ABC98].

### 2.4.1. Control predictivo de sistemas afines a tramos

Un tipo de sistemas no lineales de especial interés es el de los sistemas afines a tramos (*piecewise affine*, PWA). Los sistemas PWA se definen realizando una partición (habitualmente poliédrica) de los espacios de estados y entradas y asociando a cada región una dinámica lineal diferente:

$$x_{k+1} = A^i x_k + B^i u_k + f^i \quad \text{para} \quad \begin{bmatrix} x_k \\ u_k \end{bmatrix} \in \mathcal{C}^i$$

donde $x_k \in \mathbb{R}^{n_c} \times \{0,1\}^{n_l}$, $u_k \in \mathbb{R}^{m_c} \times \{0,1\}^{m_l}$ y $\{\mathcal{C}^i\}$ es una partición poliédrica del espacio de estados y acciones de control.
Es decir, tanto las entradas como estados pueden ser variables reales o lógicas (binarias).

Este tipo de sistemas es importante porque permite modelar de manera exacta un amplio espectro de sistemas híbridos, cuya evolución responde a ecuaciones dinámicas y reglas lógicas [HSB01]. Esto incluye multitud de procesos físicos, como sistemas lineales discretos con componentes estáticas lineales a tramos o sistemas conmutados, cuyo comportamiento dinámico está descrito por un número finito de sistemas lineales discretos junto a unas reglas lógicas para cambiar entre ellos. Además, los sistemas PWA pueden aproximar sistemas no lineales mediante la linealización en diferentes puntos de funcionamiento [Bor03].

Aunque a lo largo de los años han aparecido diferentes propuestas para el control de sistemas PWA, entre ellas destacan las que plantean un control óptimo de horizonte finito con horizonte móvil (control predictivo). El problema de optimización a resolver en este caso es el siguiente:

$$\mathcal{P}_N(x) : V_N^{OPT}(x) := \min \frac{1}{2}x_N^T P x_N + \frac{1}{2}\sum_{k=0}^{N-1}(x_k^T Q x_k + u_k^T R u_k) \tag{2.35}$$

$$\text{sujeto a:}$$

$$x_{k+1} = A^i x_k + B^i u_k + f^i \quad \text{para} \begin{bmatrix} x_k \\ u_k \end{bmatrix} \in \mathcal{C}^i; k = 0, \ldots, N-1,$$

$$x_0 = x,$$

$$x_N \in \mathbb{X}_f \subset \mathbb{X}$$

donde las restricciones sobre los estados y acciones de control ($\mathbb{U}$ y $\mathbb{X}$) están incluidas en las regiones de definición del sistema ($\mathcal{C}^i$).

La aplicación del control predictivo requiere, debido a la filosofía del horizonte móvil, la resolución del anterior problema de optimización en cada período de muestreo. Dicho problema puede ser resuelto como un problema de programación mixta entera-cuadrática (*mixed integer quadratic programming*, MIQP) [BM99].

A pesar de que existen diferentes métodos para realizar de manera eficiente la optimización anterior, como la basada en métodos Branch and Bound de [PCPC05], cuando el número de variables es elevado el coste computacional puede ser prohibitivo. Por ello, existen diferentes propuestas que tratan de obtener una solución explícita del problema de optimización, en analogía a lo realizado en la sección 2.3 para el control predictivo de sistemas lineales.

Una primera aproximación a la solución explícita del problema se propuso en [MR02]. La idea es, en primer lugar, definir todas las posibles secuencias de conmutación entre las distintas regiones $\mathcal{C}^i$ que sigue la trayectoria del sistema. Si, dado un horizonte $N$ se supone conocida una trayectoria, $s = \{s_1, \ldots, s_N\}$, el sistema puede tratarse como un sistema variante en el tiempo, que evoluciona en cada instante con una dinámica $x_{k+1} = A^{s_i}x_k + B^{s_i}u_k + f^{s_i}$. Si se exigen además de las restricciones del problema original (2.35) que la evolución del sistema sea la marcada por el sistema, $x_{k+i} \in \mathcal{C}^{s_i}$ (restricción también poliédrica), se tiene un problema de programación cuadrática multiparamétrica, cuya solución es

posible obtener.
De esta forma, para cada secuencia $s$ posible, se obtiene una caracterización del espacio de estados con un posible solución. La solución óptima a aplicar en linea se obtiene por comparación de los distintos índices de coste, lo que en general aumenta considerable el tiempo de cómputo en línea.

Otra posibilidad de obtener la solución explícita completa al problema de control óptimo con horizonte de control finito e índice de coste cuadrático puede encontrarse en [BBBM05], donde los autores muestran que la partición del espacio de estados no es en general poliédrica, pero sí que se puede definir mediante desigualdades lineales y cuadráticas. El procedimiento en este caso busca evitar la enumeración de todas las secuencias de conmutación posibles. Para ello, emplea técnicas de programación dinámica.
Por el principio de optimalidad de Hamilton-Jacobi-Bellman, se puede proceder definiendo problemas de optimización hacia atrás en el tiempo. Así, el primer paso sería resolver el siguiente problema de optimización:

$$\begin{aligned} V_1^{OPT}(x_{N-1}) :=& \min_{u_{N-1}} \frac{1}{2} x_N^T P x_N + \frac{1}{2}(x_{N-1}^T Q x_{N-1} + u_{N-1}^T R u_{N-1}) \\ & \text{sujeto a:} \\ & x_N = A^i x_{N-1} + B^i u_{N-1} + f^i \quad \text{para } \begin{bmatrix} x_{N-1} \\ u_{N-1} \end{bmatrix} \in \mathcal{C}^i \\ & x_N \in \mathbb{X}_f \subset \mathbb{X} \end{aligned}$$

Si se suponen que $(x_{N-1}, u_{N-1})$ pertenece a una determinada región $\mathcal{C}^i$, el problema anterior es un $mpQP$, del que se sabe obtener la solución. Resolviendo un problema de este tipo para cada una de las $n_s$ posibles regiones $\mathcal{C}^i$, se tienen $Nr_{N-1}$ regiones poliédricas con expresión afines de la solución del problema. No obstante, puesto que puede haber regiones que se intersecten ofreciendo más de una posible solución (multiplicidad $d$) es necesario almacenar todas las regiones en un orden adecuado para comparar los costes que producen las soluciones correspondientes a diferentes regiones $\mathcal{C}^i$. La unión de todas las regiones proporciona el conjunto de estados para el que el problema es factible, $\mathbb{X}_{N-1}$.
A continuación, puede darse un paso más y resolver el problema:

$$\begin{aligned} V_2^{OPT}(x_{N-1}) :=& \min_{u_{N-2}} V_1^{OPT}(x_{N-1}) + \frac{1}{2}(x_{N-2}^T Q x_{N-2} + u_{N-2}^T R u_{N-2}) \\ & \text{sujeto a:} \\ & x_{N-1} = A^i x_{N-2} + B^i u_{N-2} + f^i \quad \text{para } \begin{bmatrix} x_{N-2} \\ u_{N-2} \end{bmatrix} \in \mathcal{C}^i \\ & x_{N-1} \in \mathbb{X}_{N-1} \end{aligned}$$

En este caso, el coste $V_{N-2}$ no es una función cuadrática, sino cuadrática a tramos, puesto que tendrá una expresión diferente dependiendo de en cuál de las $Nr_{N-1}$ regiones se encuentre el estado $x$. Por tanto, obtener la solución del problema, requerirá la resolución de $n_s \cdot Nr_{N-1}$ mpQPs. De nuevo, la solución vendrá dada por un número de regiones poliédricas, que deben estar ordenadas adecuadamente y para las que puede haber más de una solución posible.

Si se sigue el procedimiento, finalmente se llega al problema $V_N^{OPT}(x)$, cuya solución desea obtenerse.

El problema del control predictivo de sistemas PWA se trata también en [AB06] donde se aúnan ideas de las dos propuestas anteriores. La idea es enumerar todas las secuencias de conmutación posible, como en [MR02], pero se comprueba, mediante programación dinámica, cuáles de ellas son factibles (no vacías).
Después, se resuelve un mpQP para cada secuencia factible y se intenta eliminar algunas de las regiones de las particiones resultantes. Para ello, primero se comprueba mediante un problema de programación cuadrática-entera mixta (MIQP) cuáles de las regiones son *poliedros de factibilidad única*, o lo que es lo mismo, contienen puntos para los que sólo existe una secuencia de conmutación factible. Estas regiones se conservan para la solución final. Por contra, las regiones de factibilidad múltiple pueden ser eliminadas si tienen siempre un coste superior que el asociado a las regiones con las que se solapan. Para comprobarlo, se propone un algoritmo basado en el cálculo de la diferencia de funciones convexas definidas sobre un poliedro, que resuelve QPs que van calculando cotas (superiores e inferiores) a dicha diferencia.

## 2.5. Estabilidad en control predictivo

### 2.5.1. Estabilidad del control óptimo con horizonte móvil

Para el análisis de estabilidad del control predictivo, con objeto de simplificar los resultados, se plantea el problema de optimización genérico con horizontes de control y predicción iguales ($M = N$)[1]:

$$\begin{aligned}\mathcal{P}_N(x): \quad & V_N^{OPT}(x) := \min V_N(\{x_k\},\{u_k\}),\\ & \text{sujeto a:}\\ & x_{k+1} = f(x_k,u_k) \text{ para } k=0,\dots,N-1,\\ & x_0 = x,\\ & u_k \in \mathbb{U} \text{ para } k=0,\dots,N-1,\\ & x_k \in \mathbb{X} \text{ para } k=0,\dots,N,\\ & x_N \in \mathbb{X}_f \subset \mathbb{X},\\ & V_N(\{x_k\},\{u_k\}) := F(x_N) + \textstyle\sum_{k=0}^{N-1} L(x_k,u_k)\end{aligned} \tag{2.36}$$

Como se ha visto anteriormente, la solución al problema planteado es una secuencia de acciones de control de la forma:

$$\mathscr{U}_x^{\text{opt}} := \{u_0^{\text{opt}}, u_1^{\text{opt}}, \dots, u_{N-1}^{\text{opt}}\} \tag{2.37}$$

Para obtener la ley de control a aplicar, se utiliza la filosofía del horizonte móvil, considerando únicamente las $m$ primeras acciones de control de la secuencia

[1] Para un estudio completo de la estabilidad y robustez en esquemas de control predictivo con horizonte $M \leq N$, véase [Ker00]

anterior:

$$\mathcal{K}_N(x) = u_0^{\text{opt}} \tag{2.38}$$

A lo largo de este epígrafe, se utilizarán los siguientes conceptos relativos a la estabilidad de un sistema [GS05]:

**Definición 2.2.** Sea $\mathbb{S}$ un conjunto en $\mathbb{R}^n$ que contiene al origen. Sea $f : \mathbb{R}^n \to \mathbb{R}^n$ tal que $f(\mathbb{S}) \subset \mathbb{S}$. Supóngase que el sistema

$$x_{i+1} = f(x_i), \tag{2.39}$$

con $x_i \in \mathbb{R}^n$, tiene un *punto de equilibrio* en el origen $x = 0$, es decir, $f(0) = 0$. Sea $x_0 \in \mathbb{S}$ y $\{x_i\} \subset \mathbb{S}$, $i \geq 0$, la secuencia resultante de (2.39). Se dice que el punto de equilibrio es:

1. *Estable (Lyapunov) en* $\mathbb{S}$: si para cualquier $\varepsilon > 0$, existe un $\delta > 0$ tal que
$$x_0 \in \mathbb{S} \text{ y } ||x_0|| < \delta \Rightarrow ||x_i|| < \varepsilon \text{ para todo } i \geq 0;$$
2. *Atractivo en* $\mathbb{S}$: si existe un $\eta > 0$ tal que
$$x_0 \in \mathbb{S} \text{ y } ||x_0|| < \eta \Rightarrow \lim_{i\to\infty} x_i = 0;$$
3. *Globalmente atractivo*: si
$$x_0 \in \mathbb{S} \Rightarrow \lim_{i\to\infty} x_i = 0;$$
4. *Asintóticamente estable en* $\mathbb{S}$: si es a la vez estable en $\mathbb{S}$ y atractivo en $\mathbb{S}$;
5. *Exponencialmente estable en* $\mathbb{S}$: si existen las constantes $\theta > 0$ y $\rho \in (0, 1)$ tales que
$$x_0 \in \mathbb{S} \Rightarrow ||x_i|| \leq \theta||x_0||\rho^i \text{ para todo } i \geq 0;$$

**Definición 2.3.** El conjunto $\mathbb{S}_N$ de *estados iniciales factibles* es el conjunto de estados iniciales $x \in \mathbb{X}$ para los que existen secuencias de estados y acciones de control para el problema de control óptimo con horizonte fijo (2.36).

El sistema en bucle cerrado formado por un proceso $x_{k+1} = f(x_k, u_k)$ y el controlador predictivo con restricciones (2.36)-(2.38) tiene un comportamiento no lineal, incluso cuando se controlan procesos lineales $x_{k+1} = f(x_k, u_k) = Ax_k + Bu_k$. Por dicha razón, la herramienta natural para el análisis de su estabilidad es la estabilidad de Lyapunov. Por la propia naturaleza del control predictivo, la base de la que históricamente se partió para el desarrollo de la teoría de estabilidad es la de la estabilidad del control óptimo. No obstante, el control predictivo presenta dos diferencias fundamentales con este último que dificultan el análisis: el índice de coste tiene un horizonte finito y el control en bucle cerrado aplicado utiliza la filosofía del horizonte móvil.

Para evitar estas dificultades, numerosos autores han propuesto soluciones en apariencia diferentes. No obstante, en el estudio bibliográfico que recoge el estado del arte sobre este tema de [MRRS00], los autores muestran que todas las propuestas utilizan diferentes combinaciones de tres *ingredientes*:

- Un conjunto de restricciones terminal $\mathbb{X}_f$ en el espacio de estados invariante bajo la ley de control terminal.
- Una ley de control terminal estabilizante $\mathcal{K}_f(x)$.
- Una ponderación del estado terminal $F(x)$ que habitualmente corresponde con el valor de la función de coste para el uso de la ley de control terminal con un horizonte infinito.

A partir de dichos ingredientes, las diferentes propuestas bibliográficas demuestran la estabilidad del controlador predictivo con dos enfoques diferentes:

El primero de ellos es el enfoque directo, que emplea la función de coste $V_N^{OPT}(x)$ como función de Lyapunov. Para ello, se establecen diferentes condiciones sobre $\mathbb{X}_f$, $\mathcal{K}_f(x)$ y $F(x)$ que garanticen que se cumple:

$$V_N^{OPT}(x^+) - V_N^{OPT}(x) + L(x, \mathcal{K}_N(x)) \leq 0 \tag{2.40}$$

donde $x^+ = f(x, \mathcal{K}_N(x))$. Esta es la filosofía que subyace, entre otras, a las propuestas de[CA98],[CM96], [KG88], [MM88], [MM93], [RM93] y [SR98].

El segundo enfoque se basa en la monotonicidad de la secuencia de funciones de coste $V_j^{OPT}(x)$. Estos métodos utilizan la siguiente igualdad:

$$V_N^{OPT}(x^+) - V_N^{OPT}(x) + L(x, \mathcal{K}_N(x)) = V_N^{OPT}(x^+) - V_{N-1}^{OPT}(x^+)$$

De esta forma, la desigualdad (2.40) puede demostrarse probando que la función $V_j^{OPT}(x)$ es monótona decreciente. Este enfoque es utilizado, entre otros autores, por [BCM94],[GWB90], [CS82], [DMS96], [NMS96], [MS97], [Mea97] y [PN97].

Sintetizando en un marco común todas las propuestas para la estabilidad del control predictivo, en [MRRS00] se proponen las siguientes **condiciones suficientes de estabilidad** para el problema (2.36):

**C1** La ponderación de etapa $L(x, u)$ en (2.36) satisface $L(0, 0) = 0$ y $L(x, u) \geq \gamma(||x||)$ para todo $x \in \mathbb{S}_N$, $u \in \mathbb{U}$, donde $\gamma : [0, \infty) \to [0, \infty)$ es continua, $\gamma(t) > 0$ para todo $t > 0$ y $\lim_{t\to\infty} \gamma(t) = \infty$.

**C2** La ponderación terminal del estado $F(x)$ en (2.36) satisface $F(0) = 0$, $F(x) \geq 0$ para todo $x \in \mathbb{X}_f$ y cumple la siguiente propiedad: existe una ley de control terminal $\mathcal{K}_f : \mathbb{X}_f \to \mathbb{U}$ tal que $F(f(x, \mathcal{K}_f(x))) - F(x) \leq -L(x, \mathcal{K}_f(x))$ para todo $x \in \mathbb{X}_f$.

**C3** El conjunto $\mathbb{X}_f$ es positivamente invariante para el sistema $x_{i+1} = f(x_i, u_i)$ bajo $\mathcal{K}_f(x)$, es decir, $f(x, \mathcal{K}_f(x)) \in \mathbb{X}_f$ para todo $x \in \mathbb{X}_f$.

**C4** La ley de control terminal $\mathcal{K}_f(x)$ satisface las restricciones del control en $\mathbb{X}_f$, es decir, $\mathcal{K}_f(x) \in \mathbb{U}$ para todo $x \in \mathbb{X}_f$.

**C5** Los conjuntos $\mathbb{U}$ y $\mathbb{X}_f$ contienen el origen de sus respectivos espacios.

Con las condiciones de estabilidad enunciadas, y considerando las diferentes definiciones relativas a la estabilidad propuestas anteriormente, se formula el siguiente teorema [GS05]:

**Teorema 2.1.** *Considérese el sistema*

$$x_{i+1} = f(x_i, u_i) \quad \text{para } i \geq 0, \quad f(0,0) = 0,$$

*controlado por el esquema de horizonte móvil* (2.36)-(2.38) *y supóngase que satisface las condiciones* ***C1*-*C5*** *enunciadas anteriormente. Entonces:*

1. *El conjunto* $\mathbb{S}_N$ *de estados iniciales factibles es positivamente invariante para el sistema en bucle cerrado.*
2. *El origen es globalmente atractivo en* $\mathbb{S}_N$ *para el sistema en bucle cerrado.*
3. *Si, además de* ***C1*-*C5***, *se cumple* $0 \in \text{int } \mathbb{S}_N$ *y la función de coste* $V_N^{OPT}$ *es continua en un entorno del origen, entonces el origen es asintóticamente estable en* $\mathbb{S}_N$ *para el sistema en bucle cerrado.*
4. *Si, además de* ***C1*-*C5***, *se cumple* $0 \in \text{int } \mathbb{X}_f$, $\mathbb{S}_N$ *es compacta,* $\gamma(t) \geq at^\sigma$ *en* ***C1***, $F(x) \leq b||x||^\sigma$ *para todo* $x \in \mathbb{X}_f$ *en* ***C2***, *donde* $a > 0$, $b > 0$ *y* $\sigma > 0$ *son constantes reales, y la función de coste* $V_N^{OPT}$ *es continua en* $\mathbb{S}_N$, *entonces el origen es exponencialmente estable en* $\mathbb{S}_N$ *para el sistema en bucle cerrado.*

□

**Prueba.** Véase [GS05]. ■

### 2.5.2. Estabilidad del control predictivo con modelos lineales y restricciones convexas

En esta sección, se estudiará la aplicación del teorema 2.1 al caso más sencillo y habitual de control predictivo: modelo lineal del sistema y función de coste cuadrática, para lo que se definirá la función de ponderación de etapa como $L(x_k, u_k) = \frac{1}{2}(x_k^T Q x_k + u_k^T R u_k)$. De esta forma, el problema de optimización (2.36) queda particularizado de la siguiente forma:

$$\mathcal{P}_N(x) : V_N^{OPT}(x) := \min \; F(x_N) + \frac{1}{2}\sum_{k=0}^{N-1}(x_k^T Q x_k + u_k^T R u_k) \tag{2.41}$$

sujeto a:

$x_{k+1} = Ax_k + Bu_k$ para $k = 0, \ldots, N-1$,

$x_0 = x$,

$u_k \in \mathbb{U}$ para $k = 0, \ldots, N-1$,

$x_k \in \mathbb{X}$ para $k = 0, \ldots, N$,

$x_N \in \mathbb{X}_f \subset \mathbb{X}$

En primer lugar, se comprobarán las condiciones **C1-C5** para la aplicación del teorema:

**C1:** Puede comprobarse que se cumple para una función $\gamma(t) = \lambda_{min}(Q)t^2$, donde $\lambda_{min}(Q)$ es el mínimo valor propio de la matriz $Q$.

**C2:** Para la satisfacción de esta condición, se definen la función de ponderación terminal y ley de control terminal:

$$F(x_N) = \tfrac{1}{2}x_N^T P x_N$$
$$\mathcal{K}_f = -Kx$$

Donde las matrices $P$ y $K$ puede calcularse resolviendo la ecuación algebraica de Riccati:

$$P = A^T PA + Q - K^T \bar{R} K \tag{2.42a}$$
$$K = \bar{R}^{-1} B^T PA \tag{2.42b}$$
$$\bar{R} = R + B^T PB \tag{2.42c}$$

Con esta elección, y teniendo en cuenta que $P$ es definida positiva, se tiene $F(0) = 0$ y $F(x) \geq 0$ para cualquier $x$. Por otra parte:

$$\begin{aligned}
&F(f(x, \mathcal{K}_f(x))) - F(x) + L(x, \mathcal{K}_f(x)) = \\
&= \frac{1}{2}\Big((Ax + B\mathcal{K}_f(x))^T P(Ax + B\mathcal{K}_f(x)) - x^T Px + \\
&+ x^T Qx + (\mathcal{K}_f(x))^T R(\mathcal{K}_f(x))\Big) \\
&= \frac{1}{2}x^T\Big((A - BK)^T P(A - BK) - P + Q + K^T RK\Big)x \\
&= \frac{1}{2}x^T\Big(A^T PA - P + Q + K^T(B^T PB + R)K - 2A^T PBK\Big)x \\
&= \frac{1}{2}x^T\Big(A^T PA - P + Q + (\bar{R}^{-1}B^T PA)^T B^T PA - 2A^T PB\bar{R}^{-1}B^T PA\Big)x \\
&= \frac{1}{2}x^T\Big(A^T PA - P + Q - A^T PB\bar{R}^{-1}B^T PA\Big)x \\
&= \frac{1}{2}x^T\Big(A^T PA - P + Q - A^T PB\bar{R}^{-1}\bar{R}\bar{R}^{-1}B^T PA\Big)x \\
&= \frac{1}{2}x^T\Big(A^T PA - P + Q - K^T \bar{R}K\Big)x = 0
\end{aligned}$$

**C3 y C4:** Ambas condiciones se satisfacen si se elige como $\mathbb{X}_f$ el máximo conjunto positivamente invariante para el que se satisfacen todas las restricciones en el sistema en bucle cerrado con el controlador terminal, $x_{k+1} = (A - BK)x_k$ . Dicho conjunto se conoce como *máximo conjunto admisible de salida* [GT91]:

$$\mathcal{O}_\infty = \{x | K(A - BK)^k x \in \mathbb{U} \text{ y } (A - BK)^k x \in \mathbb{X} \text{ para } k = 0, 1, \ldots\}$$

**C5:** Para el cumplimiento de esta condición, se asume desde la propia definición del problema que los conjuntos $\mathbb{U}$ y $\mathbb{X}_f$ contienen el origen de sus respectivos espacios.

Con la satisfacción de todas estas condiciones, por la aplicación del teorema 2.1, se tiene que el conjunto de estados iniciales factibles $\mathbb{S}_N$ es positivamente invariante y que con el sistema en bucle cerrado resultante de aplicar el control predictivo el origen es globalmente atractivo en $\mathbb{S}_N$. Además, puede demostrarse que, siendo $\mathbb{U}$,$\mathbb{X}$ y $\mathbb{X}_f$ conjuntos convexos, la función de coste $V_N^{OPT}(x)$ es convexa y continua [GS05], por lo que el origen es asintóticamente estable en $\mathbb{S}_N$. Por último, la elección de $F(x)$ es tal que se tiene $F(x) \leq \lambda_{max}(P)||x||^2$, siendo $\lambda_{max}(P)$ el máximo valor propio de $P$. Por tanto, queda garantizada la estabilidad exponencial.

### 2.5.3. Estabilidad del control predictivo de sistemas afines a tramos

Existen diferentes propuestas en la literatura que tratan el problema de la estabilidad de controladores predictivos en sistemas afines a tramos. El enfoque más simple es el propuesto en [Bor03], que toma como región terminal el origen, $\mathbb{X}_f = 0$. Aunque se trata de una solución que evita cálculos complejos para obtener la región terminal, tiene el importante inconveniente de que la región de estados iniciales desde los que el sistema es controlable suele ser reducida, a no ser que se empleen horizontes elevados.

Otra opción subóptima, propuesta en [BFPS09], es la de utilizar un filtro de la referencia (*reference governor*). Esta metodología, que reduce el coste computacional, utiliza una región terminal poliédrica. El inconveniente, además de que la solución obtenida es un subóptimo, es que la región de atracción es también más reducida de lo que podría ser.

Una solución menos conservativa puede conseguirse si se usa una ponderación del estado terminal $F(x)$ cuadrática a tramos y una ley de control terminal $\mathcal{K}_f$ afín a tramos, como en [LHWB06]. Aunque lo ideal en este caso es utilizar como región terminal el máximo conjunto positivamente invariante, algoritmos para su cálculo como el propuesto en [RGK+04] son bastante costosos y además en general obtienen regiones poliédricas no convexas con muchos elementos. Por ello, en [LHWB06] se proponen regiones poliédricas no convexas más sencillas, buscando un compromiso entre el tamaño de la región de atracción y la complejidad del problema resultante.

### 2.5.4. Teoría de conjuntos invariantes

La teoría de estabilidad del control predictivo expuesta hace uso de dos conjuntos poliédricos (aparte de los conjuntos de restricciones $\mathbb{X}$ y $\mathbb{U}$):

- El conjunto de restricciones terminal, $\mathbb{X}_f$.
- El conjunto de estados iniciales factibles, $\mathbb{S}_N$.

En esta sección, se describirá el procedimiento para el cálculo de ambos conjuntos. Para ello, se introducirán algunas definiciones necesarias siguiendo la terminología de [Ker00], que a su vez se apoya en el estudio bibliográfico de [Bla99].
Inicialmente, serán fundamentales los conceptos de conjuntos positivamente invariante e invariante bajo control que se introducen a continuación:

**Definición 2.4.** El conjunto $\Omega \subset \mathbb{R}^n$ es positivamente invariante para el sistema $x_{k+1} = f(x_k)$ si y solo si $\forall x_0 \in \Omega$ el sistema satisface $x_k \in \Omega$, $\forall k \in \mathbb{N}$.

En otras palabras, $\Omega$ es positivamente invariante si y sólo si

$$x_k \in \Omega \Rightarrow x_{k+1} \in \Omega$$

**Definición 2.5.** El conjunto $\Omega \subset \mathbb{R}^n$ es invariante bajo control para el sistema $x_{k+1} = f(x_k, u_k)$ si y solo si existe una ley de control $u_k = h(x_k)$ tal que $\Omega$ sea un conjunto positivamente invariante para el sistema en bucle cerrado $x_{k+1} = f(x_k, h(x_k))$ y $u_k \in \mathbb{U}, \forall x_k \in \Omega$.

Es decir, $\Omega$ es invariante bajo control si y solo si

$$x_k \in \Omega \Rightarrow \exists u_k \in \mathbb{U} : x_{k+1} \in \Omega$$

Para determinar si un conjunto es positivamente invariante o invariante bajo control, será de utilidad la siguiente definición:

**Definición 2.6.** El *conjunto a un paso* $\mathcal{Q}(\Omega)$ es el conjunto de estados en $\mathbb{R}^n$ para los que existe alguna acción de control admisible que lleve al sistema a $\Omega$ en un solo paso:

$$\mathcal{Q}(\Omega) = \{x_k \in \mathbb{R}^n | \exists u_k \in \mathbb{U} : f(x_k, u_k) \in \Omega\}$$

Dicha definición, permite formular el siguiente teorema [DH99]:

**Teorema 2.2.** *El conjunto* $\Omega \subset \mathbb{R}^n$ *es invariante bajo control si y solo si* $\Omega \subset \mathcal{Q}(\Omega)$.

### Conjunto de restricciones terminal

Como se ha visto en el epígrafe anterior, con objeto de satisfacer las condiciones de estabilidad **C3** y **C4**, es conveniente hacer el conjunto de restricciones terminal $\mathbb{X}_f$ igual al máximo conjunto invariante de entrada admisible para la ley de control $u_k = \mathcal{K}_f(x_k)$. Dicho conjunto puede entenderse a partir de los dos conceptos que se definen a continuación:

**Definición 2.7.** El conjunto $\mathcal{O}_\infty(\Omega)$ es el *máximo conjunto positivamente invariante* contenido en $\Omega$ para el sistema autónomo $x_{k+1} = f(x_k)$ si y sólo si $\mathcal{O}_\infty(\Omega)$ es positivamente invariante y contiene a todos los conjuntos positivamente invariantes contenidos en $\Omega$

**Definición 2.8.** Dada una ley de control $u_k = h(x_k)$, el *subconjunto de entrada admisible* de $\Omega \subset \mathbb{R}^n$ está dado por:

$$\Omega^h = \{x_k \in \Omega | h(x_k) \in \mathbb{U}\}$$

Por tanto, el máximo conjunto invariante de entrada admisible para la ley de control terminal se denotará como:

$$\mathcal{O}_\infty^{\mathcal{K}_f}(\Omega) = \mathcal{O}_\infty(\Omega^{\mathcal{K}_f})$$

Como ayuda para el cálculo del resto de conjuntos invariantes, es de utilidad enunciar la siguiente definición:

**Definición 2.9.** El *conjunto controlable en i pasos* $\mathbb{K}_i(\Omega, \mathbb{T})$ es el conjunto máximo de estados en $\Omega$ para los que existe una secuencia admisible de acciones de control que lleve al estado al conjunto terminal $\mathbb{T}$ en exactamente $i$ pasos, y que a la vez mantenga la evolución del estado dentro de $\Omega$ para los primeros $i-1$ pasos:

$$\mathbb{K}_i(\Omega, \mathbb{T}) = \left\{x_0 \in \mathbb{R}^n | \exists \{u_k \in \mathbb{U}\}_0^{i-1} : \{x_k \in \Omega\}_0^{i-1}, x_i \in \mathbb{T}\right\}$$

El límite de los conjuntos controlables a $i$ pasos, si existe, define el *conjunto controlable de tiempo infinito*:

$$\mathbb{K}_\infty(\Omega, \mathbb{T}) = \lim_{i \to \infty} \mathbb{K}_i(\Omega, \mathbb{T})$$

Para calcular los conjuntos controlables, se utilizará el concepto de conjunto a un paso introducido anteriormente aprovechando la siguiente propiedad:

$$\mathbb{K}_1(\Omega, \mathbb{T}) = \mathcal{Q}(\mathbb{T}) \cap \Omega$$

Con esta idea, puede desarrollarse el algoritmo 2.6 [BR71] para el cálculo de todos los conjuntos controlables.

**Algoritmo 2.6** Cálculo de conjunto controlable en $N$ pasos $\mathbb{K}_N(\Omega, \mathbb{T})$

1. Hacer $i = 0$ y $\mathbb{K}_0(\Omega, \mathbb{T}) = \mathbb{T}$
2. Mientras $i < N$:
   *a*) $\mathbb{K}_{i+1}(\Omega, \mathbb{T}) = \mathcal{Q}\left(\mathbb{K}_i(\Omega, \mathbb{T})\right) \cap \Omega$
   *b*) Si $\mathbb{K}_{i+1}(\Omega, \mathbb{T}) = \mathbb{K}_i(\Omega, \mathbb{T})$, finalizar el algoritmo y $\mathbb{K}_N(\Omega, \mathbb{T}) = \mathbb{K}_\infty(\Omega, \mathbb{T}) = \mathbb{K}_i(\Omega, \mathbb{T})$.
   *c*) i=i+1

Puede demostrarse, [Ker00], que cualquier conjunto positivamente invariante puede calcularse a partir del conjunto controlable correspondiente:

$$\mathcal{O}_i^h(\Omega) = \mathbb{K}_i(\Omega^h, \Omega^h)$$

En particular, el máximo conjunto positivamente invariante de entrada admisible, necesario para definir $\mathbb{X}_f$, puede obtenerse aplicando el algoritmo 2.6 como:

$$\mathcal{O}_\infty^h(\Omega) = \mathbb{K}_\infty(\Omega^h, \Omega^h)$$

#### Conjunto de estados iniciales factibles

Para obtener el conjunto de estados iniciales factibles $\mathbb{S}_N$, se hará uso de una nueva definición:

**Definición 2.10.** El conjunto $\mathcal{S}_i(\Omega, \mathbb{T})$ es el *conjunto estabilizable a $i$ pasos* contenido en $\Omega$ para el sistema $x_{k+1} = f(x_k, u_k)$ si y solo si $\mathbb{T}$ es un subconjunto invariante bajo control de $\Omega$ y $\mathcal{S}_i(\Omega, \mathbb{T})$ contiene todos los estados de $\Omega$ para los que existe un acción de control admisible que lleve el estado del sistema a $\mathbb{T}$ en $i$ pasos o menos, manteniendo la trayectoria del estado dentro de $\Omega$:

$$\mathcal{S}_i(\Omega, \mathbb{T}) = \{x_0 \in \mathbb{R}^n | \exists \{u_k \in \mathbb{U}\}_0^{i-1}, N \leq i : \{x_k \in \Omega\}_0^{N-1}, \\ \{x_k \in \mathbb{T} \subset \Omega\}_N^i, \mathbb{T} \subset \mathcal{Q}(\mathbb{T})\}$$

Observando la definición anterior puede comprobarse que, siempre que el conjunto terminal sea invariante bajo control, el conjunto estabilizable a $i$ pasos coincide con el conjunto controlable a $i$ pasos:

$$\mathcal{S}_i(\Omega, \mathbb{T}) = \mathbb{K}_i(\Omega, \mathbb{T})$$

Esto es debido a que, si es posible llevar al sistema en menos de $i$ pasos al conjunto $\mathbb{T}$, al ser éste invariante bajo control, podrá mantenerse el estado en esta región todos los periodos que sea necesario.

Por tanto, los conjuntos estabilizables pueden calcularse mediante el algoritmo 2.6.

El siguiente teorema pone de manifiesto la utilidad de los conjuntos estabilizables

**Teorema 2.3.** *Para el problema de control predictivo* (2.36)-(2.38)*, si la región terminal es invariante bajo control, el conjunto de estados iniciales factibles es igual al conjunto estabilizable a $N$ pasos*

$$\mathbb{S}_N = \mathcal{S}_N(\mathbb{X}, \mathbb{X}_f)$$

□

**Prueba.** Véase [Ker00] ■

## 2.6. Operaciones con poliedros

En esta sección, se introducen dos operaciones con poliedros que serán de utilidad en diferentes puntos de la tesis: la envolvente convexa de un poliedro no convexo y la minimización del número de elementos que lo definen.

### 2.6.1. Envolvente convexa

**Definición 2.11.** La envolvente convexa de un conjunto $C$ es la intersección de todos los conjuntos convexos que contienen a $C$. □

De la propia definición, resulta evidente que la envolvente convexa de cualquier conjunto es el menor conjunto convexo que contiene a $C$. Aunque el concepto de envolvente convexa es el mismo con independencia del conjunto $C$ con el que se está trabajando, la manera de calcularlo sí va a ser diferente dependiendo de las propiedades de dicho conjunto y la representación en que esté definido.

Cuando el conjunto $C$ es un poliedro convexo, su envolvente convexa es el mismo conjunto.

Cuando $C$ es un conjunto finito de puntos $\{x_1, \ldots, x_k\}$, su envolvente convexa es el conjunto de todas las combinaciones convexas de los puntos en $C$:

$$\text{conv } C = \{\lambda_1 x_1 + \ldots + \lambda_k x_k | x_i \in C, \quad \lambda_i \geq 0, \quad i = 1, \ldots, k, \quad \lambda_1 + \ldots + \lambda_k = 1\}$$

En este caso, obtener la envolvente convexa consiste en representar al conjunto como una intersección de semiespacios, o más precisamente, como un conjunto de soluciones a un sistema mínimo de desigualdades lineales. Este problema se conoce también como el problema de enumeración de facetas.
A pesar del gran interés existente en este problema en el ámbito de la geometría computacional, no puede considerarse resuelto para dimensiones generales [Fuk04, JZ04]. El estado del arte del problema en cuestión puede resumirse en que los algoritmos existentes se comportan bien para ciertos tipos de politopos y mal para otros, y que hay algunos tipos de politopos para los que ninguno de los algoritmos conocidos funciona adecuadamente [Jos02]. El principal resultado teórico obtenido para el problema es el algoritmo de Chazelle [Cha93], que proporciona un coste de peor caso en tiempo polinómico para una dimensión del problema fijada. No obstante, dicho coste depende del tamaño de la solución, y todavía puede ser bastante elevado. Una comparación de los principales métodos para el cálculo de la envolvente convexa de un conjunto de puntos puede consultarse en [AB95].

Por último, el caso particular que se desea estudiar es el del cálculo de la envolvente convexa de un poliedro no convexo, es decir, un conjunto definido como la unión de $\gamma$ poliedros convexos descritos mediante desigualdades lineales.
La solución más evidente a este problema es obtener los vértices de los $\gamma$ poliedros, agrupar todos los vértices en un nuevo conjunto y calcular la envolvente convexa de dicho conjunto de puntos con alguno de los métodos descritos en [AB95]. No obstante, esto puede ser computacionalmente costoso, puesto que, además de la obtención de la envolvente convexa de todos los puntos, es necesario obtener la enumeración de vértices de cada uno de los $\gamma$ poliedros. La enumeración de vértices de un poliedro es el problema dual al cálculo de la envolvente convexa de un conjunto de puntos, y se resuelve esencialmente igual

(ver [Fuk04]).
Para un caso general, en [Tiw07] se muestra que, a menos que $P = NP$, no existe ningún algoritmo *output sensitive* (de coste polinómico en función del tamaño de la entrada y la salida), aunque sí es posible encontrar algoritmos de este tipo para casos particulares.
Cuando los poliedros se encuentran en lo que se define como posición general, en [FLL01] se propone un algoritmo polinómico para calcular la envolvente convexa de la unión basado en búsqueda inversa.
En [Bal88], Balas propone un algoritmo polinómico para la unión de politopos cuyos sistemas lineales difieren sólo en su parte derecha.

### 2.6.2. Minimización del número de elementos en poliedros no convexos

Dado un poliedro no convexo definido como la unión de un cierto número de poliedros

$$\mathbb{X} = \bigcup_{r=0}^{\gamma} X_r$$

en esta sección se plantea obtener un nuevo conjunto de poliedros convexos $X_s$ que cumpla:

- Que la unión de los nuevos poliedros sea igual a la unión de los originales, $\mathbb{X} = \bigcup_{s=0}^{\gamma_s} X_s$
- Que el número de poliedros $X_s$, $\gamma_s$, sea mínimo.

La principal dificultad que aparece al abordar este problema es determinar si la unión de dos o más regiones es un poliedro convexo o no. Para decidirlo, se parte de la definición y el teorema siguientes, propuestos en [BFT01].

**Definición 2.12.** Dados dos poliedros $P$ y $Q$, su *envoltura* (envelope) $env(P, Q)$ está definida como la intersección de semiespacios que contienen a ambos poliedros, donde dichos semiespacios son facetas de los poliedros. □

Es decir, si los poliedros $P$ y $Q$ están definidos como:

$$\begin{aligned} P &= \{x \in \mathbb{R}^n | Ax \leq \alpha\} \\ Q &= \{x \in \mathbb{R}^n | Bx \leq \beta\} \end{aligned}$$

su envoltura se puede calcular como:

$$env(P, Q) = \{x \in \mathbb{R}^n | \bar{A}x \leq \bar{\alpha}, \bar{B}x \leq \bar{\beta}\}$$

donde $\bar{A}x \leq \bar{\alpha}$ es el subsistema de $Ax \leq \alpha$ obtenido tras eliminar todas las desigualdades no válidas para $Q$, y $\bar{B}x \leq \bar{\beta}$ se define análogamente con respecto a $P$.

**Teorema 2.4.** *Dados dos poliedros $P$ y $Q$, su unión $P \cup Q$ es convexa si y solo si $P \cup Q = env(P, Q)$* □

**Prueba.** Véase [BFT01]. ■

A partir de este teorema, en [BFT01] se proporciona una algoritmo que es capaz de decidir si la unión de $k$ regiones es convexa, y en caso afirmativo calcularla. No obstante, dicho algoritmo es computacionalmente costoso, puesto que requiere resolver $\prod_{i=1}^{k} \widetilde{m}_i$ LPs, donde $\widetilde{m}_i$ es el número de desigualdades de cada uno de los poliedros eliminadas para formar la envoltura.
En este punto es importante tener en cuenta además que el objetivo planteado no es decidir si una única unión de varios poliedros es convexa, sino encontrar el mínimo numero de poliedros convexos $X_s$ cuya unión es igual a la unión de los originales $X_r$. Por tanto, sería necesario utilizar un procedimiento en el que se comprobara inicialmente si la unión de todos los poliedros $X_r$ es convexa. Si no lo es, habría que ir probando todas las combinaciones posibles de menor número de poliedros originales, de forma que después de unidos se obtuviera el mínimo de poliedros finales. Se trata claramente de una explosión combinatoria que en el peor de los casos va a requerir un número muy elevado de LPs.
Otra posible forma de abordar el problema, en general con un coste computacional menor, es la propuesta en [GTM08]. En dicho trabajo, se aborda la reducción del número total de regiones poliédricas que forman una partición de un determinado poliedro $R$, sobre la que hay definida una función afín a tramos, mediante la unión de las regiones en las que dicha función toma un mismo valor.
Para ello, dada una disposición $A$ de hiperplanos $\{H_i\}_{i=1,\ldots,n}$ se define el vector de signos simplificado como:

$$SV_i(x) = \begin{cases} - & \text{si } a_i^T x \leq b_i \\ + & \text{si } a_i^T x \geq b_i \end{cases} \quad \text{para } i = \{1 \ldots n\}$$

A partir de dicha definición, es posible expresar cualquier poliedro como una *celda* de la disposición de planos:

$$P_m = \{x \in \mathbb{R}^d | SV(x) = m\}$$

donde el vector $m$ es la *marca* del poliedro $P_m$ en la disposición de hiperplanos $A$.
El concepto fundamental detrás de los algoritmos propuestos en [GTM08] es obtener una representación mínima de los poliedros con una misma solución afín (poliedros que denomina *blancos*), partiendo de su envoltura, que es convexa, y dividiéndola secuencialmente en poliedros usando los hiperplanos de $A$.
Si se tiene la disposición de planos y las marcas de las distintas regiones, esta operación es sencilla por las siguientes razones:

- La envoltura de varios poliedros $P_{m_i}$, y su marca $m_e$, puede obtenerse simplemente comprobando qué elementos de las respectivas marcas $m_i$ de los poliedros coinciden (- o +) y marcando como indefinidos (*) el resto (lema 12 en [GTM08]).

- Comprobar si un determinado poliedro $P_{m_i}$ se encuentra o no dentro de dicha envoltura también es sencillo, comprobando que ninguno de los elementos de $m_i$ sea diferente de los elementos de $m_e$.

- Cualquier conjunto de poliedros obtenido a partir de dividir la envoltura por un hiperplano, será por definición convexo.

A partir de esta idea, los autores proponen dos algoritmos para obtener el mínimo de regiones posibles. El primero de ellos está basado en una estrategia de ramificación y poda, encontrando siempre el mínimo número de regiones sin solapamiento cuya unión equivale a la unión de poliedros blancos originales. El segundo algoritmo, se basa en el hecho de que el problema puede reformularse como un problema de minimización de lógica booleana, para el que hay herramientas informáticas disponibles. Este segundo enfoque obtiene el mínimo número de regiones permitiendo solapamiento entre ellas (y por tanto, en general serán menos que las obtenidas en el primer caso). El coste computacional es menor para esta segunda propuesta.

## 2.7. Factibilidad de sistemas de ecuaciones y desigualdades

Considérese el sistema de ecuaciones y desigualdades (2.43):

$$\begin{cases} f_i(x) = 0, & (i = 1, \ldots, m) \\ g_i(x) \geq 0, & (i = 1, \ldots, p) \end{cases} \tag{2.43}$$

Se dice que (2.43) es factible o consistente cuando existe algún valor de $x$ que cumpla todos las restricciones $f_i(x) = 0$ y $g_i(x) \leq 0$ y no factible o inconsistente en caso contrario.

En esta sección, se presentan metodologías existentes para comprobar si (2.43) es factible o no en dos supuestos diferentes: ecuaciones y desigualdades lineales y ecuaciones y desigualdades polinómicas.

### 2.7.1. Sistemas lineales

Cuando las funciones $f_i(x)$ y $g_i(x)$ son lineales (y por tanto representan hiperplanos), el sistema cuya consistencia se desea comprobar puede escribirse como:

$$\begin{cases} L_f x - W_f = 0 \\ L_g x - W_g \leq 0 \end{cases} \tag{2.44}$$

donde $L_f$, $W_f$, $L_g$ y $W_g$ son vectores de tamaño, $[m \times n]$, $[m \times 1]$, $[p \times n]$ y $[p \times 1]$ respectivamente.

Podemos comprobar la consistencia del sistema anterior resolviendo el siguiente problema de optimización [BV04]:

$$\begin{aligned} &\min\ s \\ &\text{sujeto a:}\quad L_f x - W_f = 0 \\ &\qquad\qquad L_g x - W_g \leq \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} s \end{aligned} \tag{2.45}$$

Puesto que el problema (2.45) tiene tanto el índice de coste como las restricciones lineales, se trata de un problema de programación lineal (LP), con una solución muy eficiente.

Es evidente que el problema (2.45) siempre será factible, puesto que $s$ se puede hacer tan grande como sea necesario. En cuanto al conjunto de desigualdades original, será consistente siempre que $s \leq 0$, e inconsistente en el caso contrario (puesto que, al estar minimizando $s$, si ésta es mayor que 0 significa que no se puede encontrar ningún punto $x$ que satisfaga todas las restricciones originales). Cuando el conjunto de desigualdades es consistente, la resolución del problema nos proporciona además un punto factible, $x_f$.

A pesar de que la consistencia del conjunto de desigualdades está garantizada para un valor de $s \leq 0$, cuando se tiene $s = 0$ significa que para cualquier posible solución, alguna de las restricciones de desigualdad está activa (es decir, es una igualdad). Esto implica que región factible no es de dimensión completa.

### 2.7.2. Sistemas de ecuaciones polinómicas y suma de cuadrados

Para un caso más general, cuando el sistema (2.43) está definido por ecuaciones $f_i(x)$ y $g_i(x)$ polinómicas, se sigue la metodología propuesta en [Par00]. Para ello, inicialmente se van a definir una serie de conceptos necesarios:

Se denota como $\mathbb{R}[x]$ al conjunto de polinomios multivariable con coeficientes reales en las variables $\{x_1, \ldots, x_n\}$.

**Definición 2.13.** Un polinomio $p(x) \in \mathbb{R}[x]$ es *suma de cuadrados* (SOS) si existe una descomposición de la forma:

$$p(x) = \sum_i p_i^2(x)$$

donde $p_i(x) \in \mathbb{R}[x]$. □

Resulta evidente que cualquier polinomio suma de cuadrados es semidefinido positivo, pero dicha afirmación no es cierta a la inversa [Par00, Par03].

**Definición 2.14.** Dado un conjunto de polinomios multivariable $f_1, \ldots, f_m$ se define el *ideal* asociado a dicho conjunto como:

$$\mathbf{ideal}(f_1, \ldots, f_m) := \left\{ f \;\middle|\; f = \textstyle\sum_{i=1}^{m} t_i f_i, \quad t_i \in \mathbb{R}[x] \right\}$$

□

**Definición 2.15.** Dado un conjunto de polinomios multivariable $g_1, \ldots, g_p$ se define el *cono* asociado a dicho conjunto como:

$$\textbf{cono}(g_1, \ldots, g_p) := \left\{ g \,\middle|\, g = s_0 + \sum_i s_i g_i + \sum_{i,j} s_{ij} g_i g_j + \sum_{i,j,k} s_{ijk} g_i g_j g_k + \ldots \right\}$$

donde $s_\alpha \in \mathbb{R}[x]$ son polinomios suma de cuadrados. □

Por como se construyen los objetos algebraicos definidos, podemos observar una propiedad que cumplen cada uno de ellos, que será de utilidad posteriormente:

- Dado un punto $x$ para el que $(f_1(x) = 0, \ldots, f_m(x) = 0)$, para cualquier polinomio $t_i$ se cumple $t_i(x)f_i(x) = 0$ y por tanto, si $f \in \textbf{ideal}(f_1, \ldots, f_m)$ se cumple $f(x) = 0$.
- Dado un punto $x$ para el que $(g_1(x) \geq 0, \ldots, g_p(x) \geq 0)$, para cualquier polinomio SOS $s_i$ se cumple $s_i(x)g_i(x) \geq 0$ y por tanto, si $g \in \textbf{cono}(g_1, \ldots, g_p)$ se cumple $g(x) \geq 0$.

Con los conceptos anteriores es posible formular el siguiente teorema, originalmente propuesto por [Ste73]:

**Teorema 2.5** (Positivstellensatz)**.** *El sistema:*

$$\begin{cases} f_i(x) = 0, & (i = 1, \ldots, m) \\ g_i(x) \geq 0, & (i = 1, \ldots, p) \end{cases} \tag{2.46}$$

*es inconsistente en* $\mathbb{R}^n$ *si y solo si* $\exists F(x), G(x) \in \mathbb{R}^n$ *tales que:*

$$\begin{cases} F(x) + G(x) = -1 \\ F(x) \in \textbf{\textit{ideal}}(f_1, \ldots, f_m) \\ G(x) \in \textbf{\textit{cono}}(g_i, \ldots, g_p) \end{cases} \tag{2.47}$$

□

**Prueba.** La suficiencia de la condición (2.47) puede deducirse por reducción al absurdo. Supóngase que (2.46) es consistente, y tómese un $x_0$ que satisfaga las condiciones de dicho sistema. Por la definición de ideal, tenemos $F(x_0) = 0$, mientras que por la definición de cono, tenemos $G(x_0) \geq 0$. Por tanto, $F(x_0) + G(x_0) \geq 0$, con lo que llegamos a una contradicción con la primera condición de (2.47).
La necesidad de la condición puede consultarse en [BCR98]. ■

**Ejemplo 2.4.** *Se desea comprobar si el sistema* (2.48) *es consistente.*

$$\begin{cases} x_1^2 - x_2 + 1 = 0 \\ x_1 \geq x_2 \\ x_1 \leq 0 \end{cases} \tag{2.48}$$

*El teorema 2.5 dice que el sistema será inconsistente si y sólo si existen los polinomios* $s_0, s_1, s_2, s_{12}, t \in \mathbb{R}[x_1, x_2]$ *que cumplan:*

$$s_0 + s_1(x_1 - x_2) + s_2(-x_1) + s_{12}(x_1 - x_2)(-x_1) + t(x_1^2 - x_2 + 1) = -1$$

*siendo* $s_0, s_1, s_2$ *y* $s_{12}$ *SOS.*

*Es posible comprobar que el siguiente conjunto de polinomios satisface la condición impuesta:*

$$\begin{aligned} s_0 &= x_1^2 \\ s_1 &= 1 \\ s_2 &= 1 \\ s_{12} &= 0 \\ t &= -1 \end{aligned}$$

*Por lo tanto, el sistema* (2.48) *es inconsistente, como se puede comprobar en la figura 2.4.* ■

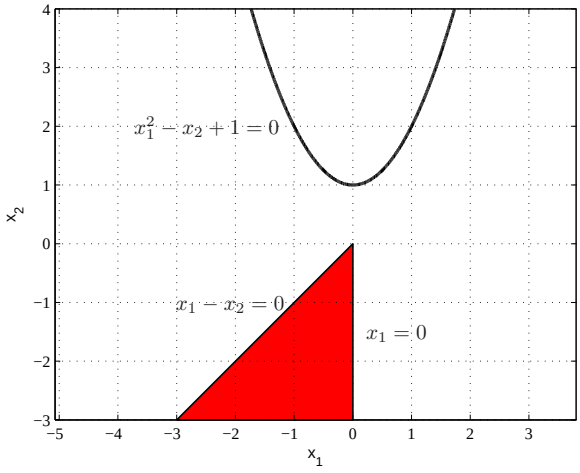


***Figura 2.4:*** Sistema inconsistente.

El teorema 2.5 proporciona alternativas excluyentes, es decir, si se encuentran polinomios $F$ y $G$ que cumplan (2.47), se tiene una prueba que refuta la consistencia de (2.46). No obstante, el teorema en sí mismo no ofrece ninguna manera de obtener dichos polinomios. Para obtener los polinomios de una manera eficiente, se hará uso del teorema 2.6, formulado en [Par03].

**Teorema 2.6.** *Dado un sistema de ecuaciones e inecuaciones polinomiales de la forma* (4.8)*, la búsqueda de refutaciones de orden acotado mediante el Positivstellensatz puede realizarse mediante programación semidefinida (SDP).* □

**Prueba.** Véase [Par03]. ■

El teorema 2.6 se basa en la idea de que obtener una refutación de (2.46) equivale a resolver el siguiente problema de factibilidad:

$$\begin{array}{ll} \text{buscar} & s_i, t_i \\ \\ \text{sujeto a:} & s_i \text{ suma de cuadrados} \\ & F(x) + G(x) = -1 \\ & F(x) = \sum_{i=1}^{m} t_i f_i(x) \\ & G(x) = s_0 + \sum_i s_i g_i(x) + \sum_{i,j} s_{ij} g_i(x) g_j(x) + \ldots \end{array} \tag{2.49}$$

Por otra parte, puede comprobarse si un polinomio $s(x)$ de un orden $2d$ determinado es suma de cuadrados expresándolo como una forma cuadrática en todos los monomios de grado igual o inferior a $d$:

$$s(x) = z^T Q z \quad z = [1, x_1, x_2, \ldots, x_n, x_1^2, x_1 x_2, \ldots, x_n^d] \tag{2.50}$$

donde $Q$ es una matriz constante.

Dado que las variables $z$ no son algebraicamente independientes la matriz $Q$ no es única, sino que hay todo un subespacio afín que satisface (2.50), pudiendo ser semidefinida positiva para algunas representaciones pero no para otras.
Si existe alguna $Q$ que, satisfaciendo (2.50), es semidefinida positiva, podemos realizar una factorización en valores propios $Q = T^T D T$ con $d_i \geq 0$ y, por tanto, podemos escribir el polinomio $s(x)$ como:

$$s(x) = z^T Q z = (Tz)^T D (Tz) = \sum_i d_i (T_i z)^2$$

que es claramente una suma de cuadrados.

En sentido inverso, si se conoce que un polinomio $s$ es suma de cuadrados de polinomios, desarrollando los términos y escribiéndolo en la forma (2.50), se obtiene una matriz $Q$ semidefinida positiva.

Es decir, comprobar si un determinado polinomio es suma de cuadrados equivale a comprobar si existe una matriz semidefinida positiva que satisfaga ciertas restricciones afines de igualdad. Dicho problema puede ser resuelto como un problema de programación semidefinida (SDP) [Par00].

En su formulación estándar, un problema SDP consiste en la minimización de un índice de coste lineal sujeto a restricciones lineales de igualdad y a que la matriz de variables sea semidefinida positiva [BV04]:

$$\begin{array}{ll} \min & \mathbf{tr}(CX) \\ \\ \text{sujeto a:} & \mathbf{tr}(A_i X) = b_i \quad i = 1, \ldots, m \\ & X \succeq 0 \end{array} \tag{2.51}$$

donde $X \in \mathbf{S}^n$ es la matriz de variables de decisión,$b \in \mathbb{R}^m$, $C, A_i \in \mathbf{S}^n$ son matrices simétricas constantes y $\mathbf{tr}(CX) = \sum_{i,j=1}^{n} C_{ij} X_{ij}$.

Una de las características fundamentales de un problema SDP es que se trata de una optimización convexa, eficiente desde un punto de vista teórico, siendo su coste de peor caso polinómico [VB96].

Además, desde un punto de vista práctico, es un problema también muy eficiente, puesto que ha sido ampliamente tratado, especialmente en problemas relacionados con la teoría de sistemas y control [BGFB94].Existen por tanto diferentes resolutores, como [Stu99], además de paquetes para la formulación eficiente de los problemas SDP, como [Lof04].

Por todo ello, es muy conveniente formular los problemas de optimización en forma de SDP siempre que sea posible.

**Ejemplo 2.5.** *Se desea comprobar si el polinomio* $F(x) = x^4+4x^3+5x^2-6x+9$ *puede escribirse como suma de cuadrados. Cómo se ha visto, esto equivale a hacer que se cumpla:*

$$F(x) = \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix}^T Q \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix} = \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix}^T \begin{bmatrix} q_{11} & q_{12} & q_{13} \\ q_{12} & q_{22} & q_{23} \\ q_{13} & q_{23} & q_{33} \end{bmatrix} \begin{bmatrix} 1 \\ x \\ x^2 \end{bmatrix} \geq 0$$

*Por tanto, hay que encontrar una matriz* $Q$ *semidefinida positiva, y que cumpla las siguientes restricciones de igualdad:*

$$\begin{aligned} q_{11} &= 9 \\ 2q_{12} &= -6 \\ 2q_{13} + q_{22} &= 5 \\ 2q_{23} &= 4 \\ q_{33} &= 1 \end{aligned}$$

*Se trata claramente de un problema de la forma* (2.51) *en el que la matriz de variables es* $Q$*, la matriz* $C$ *del índice de coste es* $[0]_{3\times 3}$ *(se trata de un problema de factibilidad y no de minimización) y las matrices* $A_i$ *se forman a partir de las restricciones de igualdad.*
*Resolviendo el SDP, se obtiene una matriz* $Q$*:*

$$Q = \begin{bmatrix} 9 & -3 & -1{,}402 \\ -3 & 7{,}804 & 2 \\ -1{,}402 & 2 & 1 \end{bmatrix}$$

*Como se ha obtenido una matriz* $Q$ *semidefinida positiva, el polinomio* $F(x)$ *puede ser escrito como suma de cuadrados. Realizando la factorización de la matriz* $Q$ *se obtiene:*

$$F(x) = (-2{,}57+2{,}20x+0{,}73x^2)^2+(-1{,}55-1{,}71x-0{,}28x^2)^2+(0{,}05-0{,}15x+0{,}62x^2)^2$$

*Como se ha comentado anteriormente, la matriz* $Q$ *que satisface las restricciones de igualdad impuestas no es única y por lo tanto pueden existir otras descomposiciones de* $F$ *como suma de cuadrados:*

$$F(x) = (x^2 + 2x)^2 + (3 - x)^2$$

■

Volviendo al problema (2.49), queda claro que, *habiendo fijado un orden para los polinomios* $s_i$, decidir si éstos son o no suma de cuadrados puede realizarse transformando el problema en un SDP. Resolver el problema en su conjunto, requiere decidir si existen polinomios $s_i$ y $t_i$ que cumplan las restricciones propuestas. Se puede observar que todas las restricciones son igualdades lineales en los parámetros (coeficientes de los polinomios), por lo que la resolución del problema puede formularse como un SDP.

La resolución del problema de optimización da lugar a dos posibles resultados:

- Si el optimizador encuentra solución, existen polinomios $F(x)$ y $G(x)$ que cumplen (2.47) por lo que, aplicando el teorema 2.5, podemos garantizar que el sistema (2.46) es inconsistente.
- Si el optimizador no encuentra solución, podemos deducir que no existen polinomios $F(x)$ y $G(x)$ que cumplan (2.47) *para un orden de los polinomios* $s_i$ *y* $t_i$ *menor o igual al fijado*. No garantiza la inexistencia de dichos polinomios para ordenes mayores, por lo que tampoco es posible descartar la inconsistencia de (2.46).

Es decir, aunque el teorema 2.5 proporciona una condición necesaria y suficiente para la inconsistencia del sistema de igualdades y desigualdades polinómicas, su implementación práctica como SDP sólo proporciona una condición suficiente.

## 2.8. Conclusiones

En la primera parte de este capítulo se ha introducido formalmente la metodología del control predictivo basado en modelos, así como algunos aspectos fundamentales asociados a ésta.
En primer lugar, usando tanto argumentos geométricos como las condiciones de optimalidad de Karush-Kuhn-Tucker, se ha mostrado que cuando el modelo de predicción es lineal y las restricciones son poliédricas, la solución al problema de optimización del control predictivo puede obtenerse de manera explícita, mediante una ley de control afín a tramos poliédricos. Además, se han introducido las diferentes propuestas bibliográficas para obtener de manera eficiente dicha solución explícita.
A continuación, se han estudiado los diferentes algoritmos existentes para la evaluación en línea de la solución explícita, haciendo especial hincapié en los árboles de búsqueda binarios, muy eficientes en términos de coste computacional.
Se ha introducido también el problema de control predictivo no lineal, brevemente en términos generales y de una manera más específica para el caso de sistemas afines a tramos, para los que se han comentado las referencias bibliográficas que muestran la existencia de una solución explícita también para este tipo de problemas.

Por último, se ha resumido el estado del arte en el campo de la estabilidad de controladores predictivos, presentando unas condiciones generales válidas para todo tipo de restricciones y modelos de predicción. Estas condiciones se han particularizado posteriormente para los casos de sistemas lineales y sistemas afines a tramos. Se han introducido también algunos conceptos de la teoría de conjuntos invariantes, útiles para las garantías de estabilidad presentadas.

Por otro lado, las secciones 2.6 y 2.7 constituyen la segunda parte del capítulo, en la que se han introducido algunas herramientas de utilidad a lo largo diferentes capítulos de la tesis enmarcadas en dos ámbitos diferentes: operaciones con poliedros y factibilidad de sistemas de ecuaciones y desigualdades.

Capítulo 3

# Control predictivo con restricciones poliédricas no convexas

## 3.1. Introducción

El control predictivo es hoy en día una tecnología asentada, con un espectro de aplicación que se ha extendido a todo tipo de procesos. Dentro de dicho espectro, pueden considerase como extremos por un lado el control predictivo lineal con restricciones poliédricas, y por el otro el control predictivo no lineal sujeto a restricciones no lineales.

En el capítulo 2 se ha estudiado el estado del arte para ambos casos. En el primero, se ha mostrado que existen diversos algoritmos eficientes en la literatura, destacando los que obtienen fuera de línea una solución explícita definida sobre regiones poliédricas, que hacen la filosofía de control predictivo factible para procesos con períodos de muestreo considerablemente bajos. En cuanto al control predictivo no lineal, se ha visto como para el caso general es necesario recurrir a algoritmos en línea de optimización no lineal (y no convexa) bastante costosos computacionalmente, por lo que sólo podrá ser aplicado a procesos que admitan períodos de muestreo relativamente altos.

En este capítulo, se plantea como alternativa intermedia a los dos casos extremos expuestos el control predictivo de sistemas lineales con índice de coste cuadrático y restricciones poliédricas no convexas, entendiendo como tal la unión no convexa de varios poliedros convexos. Este tipo de procesos, como se verá en el capítulo 7, puede utilizarse junto con algunas técnicas de control existentes como una alternativa para la resolución del control predictivo no lineal.

Como se verá a lo largo del capítulo, esta alternativa (que puede ser costosa

si se implementa mediante optimización en línea) admite una solución explícita análoga a la obtenida cuando las restricciones son poliédricas. Se estudiarán además algunas propiedades de dicha solución, que serán utilizadas en capítulos posteriores para una caracterización eficiente de la partición del espacio de estados.

## 3.2. Descripción del problema

En la sección 2.2, se definió el control predictivo como la resolución de un determinado problema de optimización (2.1) seguido de la aplicación de una estrategia de horizonte móvil. En la sección 2.2.1 se acotó el problema para un caso particular en el que se suponía un sistema lineal, un índice de coste cuadrático y restricciones descritas como desigualdades lineales. En el presente capítulo se pretende resolver un problema más general, definido por las siguientes condiciones:

- El sistema es lineal e invariante en el tiempo.
- El índice de coste es una función cuadrática.
- Las restricciones son poliedros no convexos.

**Definición 3.1.** Un *poliedro no convexo* se define como la unión no convexa de un número finito de poliedros convexos. □

En este caso, el problema de optimización genérico del control predictivo (2.1) puede formularse como:

$$\mathcal{P}_{N,M}(x) : V^{OPT}_{N,M}(x) := \min \frac{1}{2}x_N^T P x_N + \frac{1}{2}\sum_{k=0}^{N-1} x_k^T C^T Q C x_k + \frac{1}{2}\sum_{k=0}^{M-1} u_k^T R u_k \tag{3.1a}$$

sujeto a:

$$x_{k+1} = Ax_k + Bu_k \text{ para } k = 0, \ldots, N-1, \tag{3.1b}$$
$$x_0 = x,$$
$$u_k \in \bar{\mathbb{U}} \text{ para } k = 0, \ldots, M-1,$$
$$u_k = 0 \text{ para } k = M, \ldots, N-1,$$
$$x_k \in \bar{\mathbb{X}} \text{ para } k = 0, \ldots, N,$$
$$x_N \in \bar{\mathbb{X}}_f \subset \bar{\mathbb{X}}$$

donde:

$$\bar{\mathbb{U}} = \bigcup_{i=0}^{\gamma_u} \mathbb{U}_i \quad \bar{\mathbb{X}} = \bigcup_{i=0}^{\gamma_x} \mathbb{X}_i \quad \bar{\mathbb{X}}_f = \bigcup_{i=0}^{\gamma_f} \mathbb{X}_{f,i}$$

La región $\bar{\mathbb{X}}_f$ también se define como un poliedro no convexo con objeto de garantizar la estabilidad del sistema en bucle cerrado, como se discutirá en el capítulo 6.

La primera dificultad para la resolución del problema (3.1a) es que tiene restricciones sobre las variables $x_k$, distintas a las variables de optimización, $u_k$. Por tanto, es necesario utilizar el modelo de predicción que hace uso de las ecuaciones del modelo del sistema (3.1b), tal y como se hizo en la sección 2.2.1:

$$x_k = \Omega_k x + \Gamma_k \mathbf{u}$$

donde $\mathbf{u} = \{u_k, u_{k+1} \dots u_{N-1}\}$ representa la secuencia de acciones de control futuras y $\Omega_k$ y $\Gamma_k$ las filas $k$ de, respectivamente, las matrices $\Omega$ y $\Gamma$ en (2.7).

Empleando también el modelo de predicción para reescribir el índice de coste como en la sección 2.2.1, el problema de optimización puede ya escribirse en función únicamente del vector de variables de optimización $\mathbf{u}$ y el vector de parámetros medibles $x$:

$$\begin{aligned} \mathcal{P}_{N,M}(x): \quad & V_{N,M}^{OPT}(x) := \min \tfrac{1}{2}\mathbf{u}^T H \mathbf{u} + \mathbf{u}^T F x, \\ & \text{sujeto a:} \\ & u_k \in \bar{\mathbb{U}} \text{ para } k = 0, \dots, M-1, \\ & u_k = 0 \text{ para } k = M, \dots, N-1, \\ & (\Omega_k x + \Gamma_k \mathbf{u}) \in \bar{\mathbb{X}} \text{ para } k = 0, \dots, N, \\ & (\Omega_N x + \Gamma_N \mathbf{u}) \in \bar{\mathbb{X}}_f \end{aligned} \tag{3.2}$$

Una vez que todas las restricciones están ya formuladas sobre las mismas variables, una forma más compacta de expresarlas es con un único poliedro no convexo.
Para formular el problema así, al ser los conjuntos de restricciones poliedros no convexos, se enumeran todas las combinaciones de restricciones posibles. Para ello, se definen una serie de listas cuyos elementos indican qué conjuntos de restricciones son los que se están considerando:

$$\begin{aligned} c^u &= \left\{c_0^u, c_1^u, \dots, c_{M-1}^u | c_k^u \in \{1, \dots, \gamma_u\}\right\} \\ c^x &= \left\{c_0^x, c_1^x, \dots, c_{N-1}^x | c_k^x \in \{1, \dots, \gamma_x\}\right\} \\ c^f &\in \{1, \dots, \gamma_f\} \end{aligned}$$

Cada terna de listas $(c^u, c^x, c^f)$ define unívocamente un conjunto de restricciones poliédricas convexas para el problema de optimización. Por tanto, se pueden formular mediante unas matrices que permitan describir las restricciones directamente sobre el vector $(\mathbf{u}, x)$:

$$T_i := \left\{ (\mathbf{u}, x) \in \mathbb{R}^{Mm+n} \,\middle|\, \begin{bmatrix} \Phi_i & \Lambda_i \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ x \end{bmatrix} \leq \Delta_i \right\}$$

Una vez formuladas de esta forma las restricciones, el problema (3.2) puede escribirse:

$$\begin{aligned} \mathcal{P}_{N,M}(x): \quad & V_{N,M}^{OPT}(x) := \min \tfrac{1}{2}\mathbf{u}^T H \mathbf{u} + \mathbf{u}^T F x, \\ & \text{sujeto a:} \\ & (\mathbf{u}, x) \in \bar{\mathbb{T}} \end{aligned} \tag{3.3}$$

donde:

$$\bar{\mathbb{T}} = \bigcup_{i=0}^{\gamma} T_i \tag{3.4}$$

Es interesante tener en cuenta que, tal y como se han definido los conjuntos $T_i$, representan todos los posibles conjuntos de restricciones (un total de $\gamma_u^M \gamma_x^{N-1} \gamma_f$). No obstante, muchos de ellos pueden ser no factibles, representando restricciones sobre secuencias de estados y acciones de control futuras incompatibles. Con objeto de simplificar el problema de optimización a resolver, puede comprobarse de manera sencilla la factibilidad de estos conjuntos mediante la resolución de problemas de programación lineal. Una primera posibilidad es formar todos los posibles conjuntos y resolver un LP para cada uno de ellos. Otra opción es ir comprobando parcialmente la factibilidad de conjuntos de restricciones. Por ejemplo, si se comprueba que $u_k \in \mathbb{U}_1$ no es compatible con $x_{k+1} \in \mathbb{X}_1$, está combinación se elimina directamente al considerar elementos posteriores de la secuencia de control, reduciendo el número de conjuntos posibles que se van formando. Realizar esta comprobación progresiva de la factibilidad será en general menos costoso si muchas de las combinaciones se van eliminando, pero más complejo si la mayoría de los $T_i$ posibles son realmente factibles. Habitualmente, este número de $T_i$ factibles, $\gamma$, es considerablemente menor al máximo expresado anteriormente, por lo que será más conveniente dicha comprobación progresiva.

### 3.2.1. Restricciones poliédricas no convexas como sistema afín a tramos

El sistema lineal con restricciones poliédricas no convexas puede escribirse como un sistema afín a tramos como los descritos en la sección 2.4.1:

$$x_{k+1} = A^i x_k + B^i u_k + f^i \quad \text{para} \quad \begin{bmatrix} x_k \\ u_k \end{bmatrix} \in \mathcal{C}^i$$

Para ello, basta con tomar $A^i = A$, $B^i = B$,$f^i = 0$ $\forall i$ y definir las regiones $\mathcal{C}^i$ como la intersección de todas las posibles regiones $\mathbb{U}_i$ y $\mathbb{X}_i$.
Esta formulación, permitiría utilizar alguna de las metodologías expuestas en la sección 2.4.1 para la obtención de la solución explícita del control predictivo para este tipo de sistemas. No obstante, existe una particularidad fundamental en el problema con restricciones no convexas que no se cumple para el caso de los sistemas afines a tramos: la función de coste en función de $x$ y $\mathbf{u}$ es cuadrática en lugar de cuadrática a tramos.
Este hecho, como se verá a lo largo de la tesis, permite obtener resultados y desarrollar algoritmos considerablemente más eficientes, lo que es crucial en sistemas de este tipo en los que el tiempo de cómputo es una restricción de gran importancia.

## 3.3. Optimización en línea

La solución más directa al problema (3.3), se obtiene al observar que éste puede ser dividido en $\gamma$ problemas de programación cuadrática convexos de la forma (2.9):

$$\mathcal{P}_{N,M}(x): \quad V_{N,M}^{OPT}(x) := \min \left\{V_{iN}^{OPT}(x)\right\}$$

donde:

$$V_{iN}^{OPT}(x) := \min \tfrac{1}{2}\mathbf{u}^T H\mathbf{u} + \mathbf{u}^T Fx, \tag{3.5}$$

sujeto a:
$(\mathbf{u}, x) \in T_i$

Los problemas (3.5) pueden resolverse en línea con los métodos de programación cuadrática habituales. No obstante, la solución obtenida para cada uno de ellos, será el óptimo *sujeto a que las acciones de control y/o estados futuros se encuentren sucesivamente en las regiones indicadas por* $(c^u, c^x, c^f)$ . Si se quiere obtener el óptimo considerando todo el espacio de restricciones, el mínimo puede obtenerse comparando el valor de los índices de coste obtenidos.

De la expresión del valor máximo de $\gamma$, $\gamma_{max} = \gamma_u^M \gamma_x^{N-1} \gamma_f$, puede deducirse que, a medida que crecen tanto el número de regiones en las que se dividen los diferentes espacios de restricciones, como especialmente los horizontes de control y predicción, $M$ y $N$, el número de problemas de programación cuadrática a resolver se incrementa de manera muy rápida. Es lo que se conoce como *explosión combinatoria*.
Por tanto, si se pretende aplicar algún controlador de este tipo a procesos con periodo de muestreo relativamente bajo, en general no será viable la solución de resolver en línea todos los problemas de optimización que aparecen y comparar los índices de coste. Por ello, es necesario buscar soluciones alternativas.

**Ejemplo 3.1.** *Supóngase que se desea aplicar un control predictivo sobre un proceso de una entrada y una salida que presenta las siguientes restricciones sobre la acción de control:*

$$\bar{\mathbb{U}} = \mathbb{U}_1 \cup \mathbb{U}_2$$

*donde:*

$$\mathbb{U}_1 = \{u \in \mathbb{R} | -1 \leq u \leq 0\} \quad \mathbb{U}_2 = \{u \in \mathbb{R} | 1 \leq u \leq 2\}$$

*Las restricciones descritas forman un poliedro no convexo.*
*Si se toma un horizonte de control $M = 1$, es necesario formar únicamente dos listas $c^u$ (que corresponden con exigir o bien $\{-1 \leq u_k \leq 0\}$ o bien $\{1 \leq u_k \leq 2\}$):*

$$c_1^u = \{1\}$$
$$c_2^u = \{2\}$$

*Para este caso, $M = 1$, se tiene $\mathbb{T}_1 = \mathbb{U}_1$ y $\mathbb{T}_2 = \mathbb{U}_2$.*
*Si se aumenta el horizonte de control, por ejemplo a $M = 3$, las posibles listas*

*$c^u$ son $2^3 = 8$:*

$$\begin{aligned} c_1^u &= \{1,1,1\} \\ c_2^u &= \{1,1,2\} \\ c_3^u &= \{1,2,1\} \\ c_4^u &= \{1,2,2\} \\ c_5^u &= \{2,1,1\} \\ c_6^u &= \{2,1,2\} \\ c_7^u &= \{2,2,1\} \\ c_8^u &= \{2,2,2\} \end{aligned}$$

*A partir de dichas listas y de $\mathbb{U}_i$, se pueden formas las matrices correspondientes a $\mathbb{T}_i$. A modo de ejemplo, se muestra $\mathbb{T}_1$:*

$$\mathbb{T}_1 := \left\{ u \in \mathbb{R}^3 | \begin{bmatrix} 1 & 0 & 0 \\ -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & -1 \end{bmatrix} u \leq \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} \right\}$$

*Es decir, para un caso tan simple como el expuesto, en el que la zona de restricciones no convexa está formada únicamente por 2 subconjuntos convexos, para un horizonte de control bajo ($M = 3$) ya sería necesario resolver 8 problemas de programación cuadrática, con el consiguiente coste computacional que ello supone.* ■

## 3.4. Solución explícita

Vistas las dificultades en términos de coste computacional que supone resolver el problema de optimización (3.3) en línea, la alternativa que se plantea es obtener una solución explícita de dicho problema análoga a la mostrada en la sección 2.3 para el problema (2.4).
La idea para obtener dicha solución es seguir la siguiente metodología:

1. Obtener la solución explícita de cada subproblema de (3.5). De esta forma, se obtienen $\gamma$ divisiones del espacio de estados, cada una de ellas con $N_{ri}$ regiones.

2. Superponer dichas divisiones del espacio de estados. Obtendremos cómo máximo $N_r = \prod_{i=1}^{\gamma} N_{ri}$ regiones, cada una de las cuales tiene $\gamma$ posibles soluciones. No obstante, habitualmente muchas de esas regiones estarán vacías y no será necesario considerarlas.

3. Dividir en diferentes subregiones cada una de las $N_r$ nuevas regiones, en función de para cual de los subproblemas de (3.5) se obtiene el óptimo. El número máximo de subregiones en las que habrá que dividir cada una de las $N_r$ regiones es el número de subproblemas de (3.5),$\gamma$.

La implementación de una solución explícita de esta forma resulta prometedora frente a la resolución de los diferentes QPs fundamentalmente en dos sentidos:

- Como se comentó en el capítulo 2, la implementación en línea de la solución explícita de (2.4) es, al menos para algunos tamaños de problema, más eficiente que la resolución de un problema de programación cuadrática. Por tanto, la implementación en línea de la expresión explícita de $\gamma$ problemas de optimización será también más rápida que resolver $\gamma$ problemas QP.

- La implementación en línea de solución explícita de (3.3), cómo se verá en el capítulo 5, es más eficiente que la implementación en línea de las $\gamma$ soluciones explícitas de los QPs, al menos si se emplean técnicas de árbol binario como las vistas en el capítulo 2. La explicación intuitiva de este hecho es la siguiente: como se vio en dicho capítulo, el coste de los árboles binarios es logarítmico con el número de regiones, por lo que es igual de costoso recorrer un árbol binario de una partición de $N_r$ regiones que los $\gamma$ árboles de las $N_{ri}$. Como en general el numero de regiones factibles tras intersectar las regiones será mucho menor que $N_r$, el coste se verá reducido. Además, si se obtienen $\gamma$ soluciones explícitas diferentes, será necesario comparar el coste cuadrático de todas ellas, mientras que con la solución explícita total, como se verá, el número de costes cuadráticos a calcular puede ser mucho menor.

### 3.4.1. Caracterización de la solución explícita

En esta sección, utilizando argumentos geométricos similares a los expuestos en la sección 2.3, se buscará obtener una solución explícita al problema de optimización (3.3), entendiendo como tal una ley de control de realimentación del estado definida a tramos. Para ello, se introduce la siguiente definición y se formula el lema 3.1, que se utilizarán para formular el resultado principal de esta sección, el teorema 3.2.

**Definición 3.2.** Dado un conjunto de desigualdades lineales sobre una determinada variable $\mathbf{u}$ :

$$S := \{\mathbf{u} \in \mathbb{R}^n | R\mathbf{u} \leq r\} \tag{3.6}$$

Se define una cara activa de $S$ cómo un subconjunto de $S$ para el que parte de esas desigualdades son igualdades y el resto desigualdades estrictas:

$$f := \left\{ \mathbf{u} \in \mathbb{R}^n | \begin{array}{c} R_l\mathbf{u} = r_l \\ R_s\mathbf{u} < r_s \end{array} \right\} \tag{3.7}$$

□

Nótese por tanto que la cara activa es el subconjunto del hiperplano $R_l\mathbf{u} = r_l$ que satisface las desigualdades $R_s\mathbf{u} < r_s$.

**Lema 3.1.** *Considérense dos hiperplanos definidos como*

$$f_1 := \{\mathbf{u} \in \mathbb{R}^n | R_1\mathbf{u} = r_1\}$$
$$f_2 := \{\mathbf{u} \in \mathbb{R}^n | R_2\mathbf{u} = r_2\}$$

*El subconjunto de* $\mathbf{u}$ *cuyos puntos tienen una distancia euclídea igual a ambos hiperplanos está definido por la forma cuadrática:*

$$\mathbf{u}^T \left(R_1^\dagger R_1 - R_2^\dagger R_2\right) \mathbf{u} - 2 \left(R_1^\dagger r_1 - R_2^\dagger r_2\right)^T \mathbf{u} + \\ + r_1^T (R_1 R_1^T)^{-1} r_1 - r_2^T (R_2 R_2^T)^{-1} r_2 = 0 \quad (3.8)$$

*donde* $M^\dagger = M^T(MM^T)^{-1}$ *representa la pseudoinversa de una matriz* $M^{m\times n}$ *de rango* $m$. □

**Prueba.** Dado un hiperplano:

$$f_1 := \{\mathbf{u} \in \mathbb{R}^n | R_1\mathbf{u} = r_1\}$$

La proyección de un punto $\mathbf{u}_0$ sobre $f_1$ viene dada por:

$$\text{proy}(\mathbf{u}_0, f_1) = (I - R_1^\dagger R_1)\mathbf{u}_0 + R_1^\dagger r_1$$

Por definición, la distancia entre un punto y un hiperplano es igual a la distancia entre el punto y su proyección sobre dicho hiperplano:

$$\begin{aligned} \text{dist}(\mathbf{u}, f_1) &= \text{dist}\left(\mathbf{u}, \text{proy}(\mathbf{u}, f_1)\right) = \|\mathbf{u} - (I - R_1^\dagger R_1)\mathbf{u} - R_1^\dagger r_1\| \\ &= \|R_1^\dagger \left(R_1\mathbf{u} - r_1\right)\| \end{aligned}$$

Se desea obtener el conjunto de $\mathbf{u}$ para los que $\text{dist}(\mathbf{u}, f_1) = \text{dist}(\mathbf{u}, f_2)$. Por tanto, debe cumplirse:

$$\|R_1^\dagger \left(R_1\mathbf{u} - r_1\right)\| = \|R_2^\dagger \left(R_2\mathbf{u} - r_2\right)\|$$

La igualdad de normas cuadráticas de ambos vectores, es equivalente a la igualdad de productos de los vectores transpuestos por sí mismos:

$$\left(R_1^\dagger \left(R_1\mathbf{u} - r_1\right)\right)^T R_1^\dagger \left(R_1\mathbf{u} - r_1\right) = \left(R_2^\dagger \left(R_2\mathbf{u} - r_2\right)\right)^T R_2^\dagger \left(R_2\mathbf{u} - r_2\right)$$
$$\left(R_1\mathbf{u} - r_1\right)^T \left(R_1^\dagger\right)^T R_1^\dagger \left(R_1\mathbf{u} - r_1\right) = \left(R_2\mathbf{u} - r_2\right)^T \left(R_2^\dagger\right)^T R_2^\dagger \left(R_2\mathbf{u} - r_2\right)$$

Dicha expresión puede simplificarse teniendo en cuenta que:

$$\left(R_1^\dagger\right)^T R_1^\dagger = \left(R_1^T (R_1 R_1^T)^{-1}\right)^T R_1^T (R_1 R_1^T)^{-1} = (R_1 R_1^T)^{-1}$$

Por tanto:

$$\left(R_1\mathbf{u} - r_1\right)^T (R_1 R_1^T)^{-1} \left(R_1\mathbf{u} - r_1\right) = \left(R_2\mathbf{u} - r_2\right)^T (R_2 R_2^T)^{-1} \left(R_2\mathbf{u} - r_2\right)$$

Desarrollando la transpuesta de la suma y agrupando términos, se obtiene:

$$\mathbf{u}^T \left(R_1^T(R_1R_1^T)^{-1}R_1 - R_2^T(R_2R_2^T)^{-1}R_2\right)\mathbf{u} - 2\left(R_1^T(R_1R_1^T)^{-1}r_1 \right.$$
$$\left. -R_2^T(R_2R_2^T)^{-1}r_2\right)^T \mathbf{u} + r_1^T(R_1R_1^T)^{-1}r_1 - r_2^T(R_2R_2^T)^{-1}r_2 = 0$$

Por último, teniendo en cuenta de nuevo la definición de pseudoinversa:

$$\mathbf{u}^T \left(R_1^\dagger R_1 - R_2^\dagger R_2\right)\mathbf{u} - 2\left(R_1^\dagger r_1 - R_2^\dagger r_2\right)^T \mathbf{u} +$$
$$+ r_1^T(R_1R_1^T)^{-1}r_1 - r_2^T(R_2R_2^T)^{-1}r_2 = 0$$

■

**Teorema 3.2.** *Considérese el problema de optimización* (3.3)*. Se puede expresar la solución como una función explícita afín a tramos*

$$\mathbf{u}^{opt}(x) = G_i x + h_i, \quad \textit{para } x \in X_i, \quad i = 0, \ldots, N_r \tag{3.9}$$

*donde $X_i$ es una partición del espacio de estados de la forma:*

$$X_i := \left\{ x \in \mathbb{R}^n | \begin{array}{c} x^T M_{ij} x + N_{ij} x + C_{ij} \leq 0 \quad j = 1 \ldots (\gamma - 1) \\ L_i x + W_i \leq 0 \end{array} \right\} \tag{3.10}$$

□

**Prueba.** En primer lugar, se tratará el caso más sencillo posible, en que el espacio de restricciones no convexo $\overline{\mathbb{T}}$ está formado por la unión de dos regiones convexas $T_1$ y $T_2$ ($\gamma = 2$).
Tal y como se vio en la sección 3.3, es posible reformular el problema (3.3) como (3.5), obteniendo dos problemas de programación cuadrática, uno para cada una de las dos regiones:

$$\begin{array}{ll} \mathcal{P}_{1N}(x): & V_{1N}^{OPT}(x) := \min \ \frac{1}{2}\mathbf{u}^T H \mathbf{u} + \mathbf{u}^T F x, \\ & \\ & \text{sujeto a:} \\ & (\mathbf{u}, x) \in T_1 \\ \mathcal{P}_{2N}(x): & V_{2N}^{OPT}(x) := \min \ \frac{1}{2}\mathbf{u}^T H \mathbf{u} + \mathbf{u}^T F x, \\ & \\ & \text{sujeto a:} \\ & (\mathbf{u}, x) \in T_2 \end{array}$$

Realizando el mismo cambio de variable propuesto en la sección 2.3, $\widetilde{\mathbf{u}} = H^{1/2}\mathbf{u}$, los problemas de optimización pueden ser reformulados como:

$$\begin{array}{ll} \mathcal{P}_{1N}(x): & V_{1N}^{OPT}(x) := \min \ \frac{1}{2}\widetilde{\mathbf{u}}^T \widetilde{\mathbf{u}} + \widetilde{\mathbf{u}}^T H^{-1/2} F x, \\ & \\ & \text{sujeto a:} \\ & (\widetilde{\mathbf{u}}, x) \in \widetilde{T}_1 \\ \mathcal{P}_{2N}(x): & V_{2N}^{OPT}(x) := \min \ \frac{1}{2}\widetilde{\mathbf{u}}^T \widetilde{\mathbf{u}} + \widetilde{\mathbf{u}}^T H^{-1/2} F x, \\ & \\ & \text{sujeto a:} \\ & (\widetilde{\mathbf{u}}, x) \in \widetilde{T}_2 \end{array}$$

Operando como se describe en dicha sección, es posible escribir la solución explícita de estos problemas como:

$$\mathcal{K}_{1N}(x) = G_{1i}x + h_{1i}, \quad \text{para } x \in X_{1i}, \quad i = 0, \dots, N_{r1}$$
$$X_{1i} := \{x \in \mathbb{R}^n | L_{1i}x \leq W_{1i}\}$$

$$\mathcal{K}_{2N}(x) = G_{2j}x + h_{2j}, \quad \text{para } x \in X_{2j}, \quad j = 0, \dots, N_{r2}$$
$$X_{2j} := \{x \in \mathbb{R}^n | L_{2j}x \leq W_{2j}\}$$

Intersectando las particiones en las que son válidas ambas soluciones, se obtiene una nueva partición lineal del espacio de estados de, a lo sumo, $N_{r1} \cdot N_{r2}$ regiones:

$$X_r = \left\{ x \in \mathbb{R}^n \left| \begin{bmatrix} L_{1i} \\ L_{2j} \end{bmatrix} x \leq \begin{bmatrix} W_{1i} \\ W_{2j} \end{bmatrix} \right. \right\}, \quad r = 0 \dots N_{r1} \cdot N_{r2}$$

donde $r = (N_{r1} + 1) \cdot j + i$ es el índice que define la nueva partición.
En cada una de las regiones $X_r$, existen dos posibles soluciones al problema de optimización, las soluciones a los problemas $\mathcal{P}_{1N}(x)$ y $\mathcal{P}_{2N}(x)$. De las dos, el óptimo solución al problema original es aquel que produce un menor índice de coste $V_{iN}^{OPT}$:

$$\mathcal{K}_N(x) = \begin{cases} G_{1i}x + h_{1i} \text{ si } V_{1N}^{OPT}(x) < V_{2N}^{OPT}(x) & \text{para } x \in X_r, \\ G_{2j}x + h_{2j} \text{ si } V_{2N}^{OPT}(x) < V_{1N}^{OPT}(x) & r = 0, \dots, N_{r1} \cdot N_{r2} \end{cases} \tag{3.11}$$

Por tanto, el óptimo solución del problema puede expresarse como una función de $x$ afín a tramos. Queda por obtener de forma explícita una división de $X_r$ en función de cuál de las dos soluciones dé un índice de coste menor.
Para ello, se recuerda la interpretación geométrica de la solución explícita del problema de optimización $\mathcal{P}_{1N}(x)$ que se dio en la sección 2.3: la función afín de $x$ que describe la solución del problema en una determinada región $X_{1i}$ corresponde con la proyección del óptimo sin restricciones, $\widetilde{\mathbf{u}}_{uc}^{opt}(x)$, sobre la cara activa correspondiente del conjunto de restricciones $\widetilde{T}_1$:

$$f_1 := \{\widetilde{\mathbf{u}} \in \mathbb{R}^n \left| \widetilde{\Phi}_{1l}\widetilde{\mathbf{u}} = \Delta_{1l} - \Lambda_{1l}x = \Theta_{1l} \right.\}$$

Por otra parte, debido al espacio de coordenadas que se obtiene tras realizar el cambio de variable propuesto, el valor del índice de coste para el óptimo corresponde con la suma entre el valor de dicho índice para el óptimo sin restricciones y la distancia euclídea entre dicho óptimo sin restricciones y la correspondiente cara activa:

$$V_{1N}^{OPT}(\widetilde{\mathbf{u}}^{opt}) = V_{1N}^{OPT}(\widetilde{\mathbf{u}}_{uc}^{opt}) + \text{dist}(\widetilde{\mathbf{u}}^{opt}, \widetilde{\mathbf{u}}_{uc}^{opt}) = V_{1N}^{OPT}(\widetilde{\mathbf{u}}_{uc}^{opt}) + \text{dist}(\widetilde{\mathbf{u}}_{uc}^{opt}, f_1)$$

Análogamente, es posible analizar la solución de $\mathcal{P}_{2N}(x)$ representando la cara activa de $\widetilde{T}_2$ asociada a una región $X_{2j}$ y el índice de coste para el óptimo correspondiente a proyectar sobre dicha cara:

$$f_2 := \{\widetilde{\mathbf{u}} \in \mathbb{R}^n \left| \widetilde{\Phi}_{2l}\widetilde{\mathbf{u}} = \Delta_{2l} - \Lambda_{2l}x = \Theta_{2l} \right.\}$$
$$V_{2N}^{OPT}(\widetilde{\mathbf{u}}^{opt}) = V_{2N}^{OPT}(\widetilde{\mathbf{u}}_{uc}^{opt}) + \text{dist}(\widetilde{\mathbf{u}}^{opt}, \widetilde{\mathbf{u}}_{uc}^{opt}) = V_{2N}^{OPT}(\widetilde{\mathbf{u}}_{uc}^{opt}) + \text{dist}(\widetilde{\mathbf{u}}_{uc}^{opt}, f_2)$$

Dado que la expresión del índice de coste cuadrático en función de $x$ y $\mathbf{u}$ es la misma para ambos problemas, la única diferencia entre los dos problemas de programación cuadrática es el espacio de restricciones, y por tanto el valor del índice de coste para el óptimo sin restricciones es igual en ambos casos($V_{1N}^{OPT}(\widetilde{\mathbf{u}}_{uc}^{opt}) = V_{2N}^{OPT}(\widetilde{\mathbf{u}}_{uc}^{opt})$). De esta forma, es posible escribir el lugar geométrico en que el índice de coste asociado al óptimo solución de $\mathcal{P}_{1N}(x)$ es menor que el asociado al óptimo solución de $\mathcal{P}_{2N}(x)$:

$$Q_{r1} := \left\{ x \in \mathbb{R}^n \,\middle|\, V_{1N}^{OPT}(x) < V_{2N}^{OPT}(x) \right\} = \left\{ x \in \mathbb{R}^n | \mathrm{dist}(\widetilde{\mathbf{u}}_{uc}^{opt}, f_1) < \mathrm{dist}(\widetilde{\mathbf{u}}_{uc}^{opt}, f_2) \right\}$$

Teniendo presente el lema 3.1:

$$Q_{r1} := \left\{ x \in \mathbb{R}^n \,\middle|\, (\widetilde{\mathbf{u}}_{uc}^{opt})^T \left( \widetilde{\Phi}_{1l}^{\dagger} \widetilde{\Phi}_{1l} - \widetilde{\Phi}_{2l}^{\dagger} \widetilde{\Phi}_{2l} \right) \widetilde{\mathbf{u}}_{uc}^{opt} - 2 \left( \widetilde{\Phi}_{1l}^{\dagger} \Theta_{1l} - \widetilde{\Phi}_{2l}^{\dagger} \Theta_{2l} \right)^T \widetilde{\mathbf{u}}_{uc}^{opt} + \right.$$
$$\left. + \Theta_{1l}^T (\widetilde{\Phi}_{1l} \widetilde{\Phi}_{1l}^T)^{-1} \Theta_{1l} - \Theta_{2l}^T (\widetilde{\Phi}_{2l} \widetilde{\Phi}_{2l}^T)^{-1} \Theta_{2l} < 0 \right\}$$

Substituyendo $\Theta_{1l}$ y $\Theta_{2l}$ y haciendo uso de la expresión del óptimo sin restricciones $\widetilde{\mathbf{u}}_{uc}^{opt}(x) = -H^{-1/2} F x$, es posible obtener la forma cuadrática $Q_{r1}$ de forma explícita:

$$Q_{r1} := \left\{ x \in \mathbb{R}^n \,\middle|\, x^T M_{r1} x + N_{r1} x + C_{r1} \geq 0 \right\} \tag{3.12}$$

donde:

$$M_{r1} = F^T H^{-1/2} \left( \widetilde{\Phi}_{1l}^{\dagger} \widetilde{\Phi}_{1l} - \widetilde{\Phi}_{2l}^{\dagger} \widetilde{\Phi}_{2l} \right) H^{-1/2} F - 2 \left( \widetilde{\Phi}_{1l}^{\dagger} \Lambda_{1l} - \widetilde{\Phi}_{2l}^{\dagger} \Lambda_{2l} \right)^T H^{-1/2} F +$$
$$+ \Lambda_{1l}^T (\widetilde{\Phi}_{1l} \widetilde{\Phi}_{1l}^T)^{-1} \Lambda_{1l} - \Lambda_{2l}^T (\widetilde{\Phi}_{2l} \widetilde{\Phi}_{2l}^T)^{-1} \Lambda_{2l}$$
$$N_{r1} = 2 \left( \left( \widetilde{\Phi}_{1l}^{\dagger} \Delta_{1l} - \widetilde{\Phi}_{2l}^{\dagger} \Delta_{2l} \right)^T H^{-1/2} F - \Delta_{1l}^T (\widetilde{\Phi}_{1l} \widetilde{\Phi}_{1l}^T)^{-1} \Lambda_{1l} + \Delta_{2l}^T (\widetilde{\Phi}_{2l} \widetilde{\Phi}_{2l}^T)^{-1} \Lambda_{2l} \right)$$
$$C_{r1} = \Delta_{1l}^T (\widetilde{\Phi}_{1l} \widetilde{\Phi}_{1l}^T)^{-1} \Delta_{1l} - \Delta_{2l}^T (\widetilde{\Phi}_{2l} \widetilde{\Phi}_{2l}^T)^{-1} \Delta_{2l}$$

La forma cuadrática $Q_{r1}$ divide la región $X_r$ en dos partes de forma que en una de ellas el óptimo solución al problema $\mathcal{P}_{1N}(x)$ ofrece un índice de coste menor que la solución al problema $\mathcal{P}_{2N}(x)$ y en la otra ocurre lo contrario.
Por tanto la solución global al problema $\mathcal{P}_N(x)$ en la región $X_r$ se puede expresar cómo:

$$\mathcal{K}_N(x) = \begin{cases} G_{1i} x + h_{1i} \text{ para } x \in X_{r1} \\ G_{2j} x + h_{2j} \text{ para } x \in X_{r2} \end{cases}$$

donde:

$$X_{r1} = \left\{ x \in \mathbb{R}^n \,\middle|\, \begin{array}{c} x^T [-M_{r1}] x - N_{r1} x - C_{r1} \leq 0 \\ \begin{bmatrix} L_{1i} \\ L_{2j} \end{bmatrix} x \leq \begin{bmatrix} W_{1i} \\ W_{2j} \end{bmatrix} \end{array} \right\}$$
$$X_{r2} = \left\{ x \in \mathbb{R}^n \,\middle|\, \begin{array}{c} x^T M_{r1} x + N_{r1} x + C_{r1} \leq 0 \\ \begin{bmatrix} L_{1i} \\ L_{2j} \end{bmatrix} x \leq \begin{bmatrix} W_{1i} \\ W_{2j} \end{bmatrix} \end{array} \right\}$$

Trabajando de la manera descrita para todas las posibles regiones $X_r$, o lo que es lo mismo, para todas las posibles combinaciones de caras activas de

los conjuntos de restricciones $\widetilde{T}_1$ y $\widetilde{T}_2$, se obtiene una partición del espacio de estados en la que cada región estará definida por restricciones lineales, obtenidas de la intersección de las regiones que formaban las particiones anteriores ($X_{1i}$ y $X_{2j}$), y una única curva cuadrática, que divide las regiones definidas por las restricciones lineales en dos. Por tanto, las regiones puede escribirse de la forma (3.10) (con $\gamma - 1 = 1$).

A continuación, se estudia el caso de $\gamma = 3$. Para ello, se realizará de nuevo una división del problema de optimización no convexo en varios problemas de optimización convexos:

$$\mathcal{P}_N(x): \quad V_N^{OPT}(x) := \min \left\{ V_{1N}^{OPT}(x), V_{2N}^{OPT}(x), V_{3N}^{OPT}(x) \right\}$$

donde:

$$V_{iN}^{OPT}(x) := \min \tfrac{1}{2}\widetilde{\mathbf{u}}^T\widetilde{\mathbf{u}} + \widetilde{\mathbf{u}}^T H^{-1/2} F x,$$
sujeto a:
$$(\widetilde{\mathbf{u}}, x) \in \widetilde{T}_i$$

El problema anterior, puede reescribirse con sólo dos subproblemas de optimización de la siguiente forma:

$$\mathcal{P}_N(x): \quad V_N^{OPT}(x) := \min \left\{ V_{1N}^{OPT}(x), V_{23N}^{OPT}(x) \right\}$$

donde:

$$V_{1N}^{OPT}(x) := \min \tfrac{1}{2}\widetilde{\mathbf{u}}^T\widetilde{\mathbf{u}} + \widetilde{\mathbf{u}}^T H^{-1/2} F x,$$
sujeto a:
$$(\widetilde{\mathbf{u}}, x) \in \widetilde{T}_1$$

$$V_{23N}^{OPT}(x) := \min \{ V_{2N}^{OPT}(x), V_{3N}^{OPT}(x) \}$$

$$V_{2N}^{OPT}(x) := \min \tfrac{1}{2}\widetilde{\mathbf{u}}^T\widetilde{\mathbf{u}} + \widetilde{\mathbf{u}}^T H^{-1/2} F x,$$
sujeto a:
$$(\widetilde{\mathbf{u}}, x) \in \widetilde{T}_2$$

$$V_{3N}^{OPT}(x) := \min \tfrac{1}{2}\widetilde{\mathbf{u}}^T\widetilde{\mathbf{u}} + \widetilde{\mathbf{u}}^T H^{-1/2} F x,$$
sujeto a:
$$(\widetilde{\mathbf{u}}, x) \in \widetilde{T}_3$$

Las soluciones de estos dos nuevos problemas sí puede ser expresada de forma explícita. El primero de ellos es un problema con restricciones lineales, tal como los descritos en la sección 2.3 y el segundo es un problema con restricciones descritas como la unión de dos regiones convexas, para el que se acaba de demostrar la existencia de solución explícita:

$$\mathcal{K}_{1N}(x) = G_{1i}x + h_{1i}, \quad \text{para } x \in X_{1i}, \quad i = 0, \ldots, N_{r1}$$
$$X_{1i} := \{ x \in \mathbb{R}^n \,|\, L_{1i}x \leq W_{1i} \}$$

$$\mathcal{K}_{23N}(x) = G_{23r}x + h_{23r}, \quad \text{para } x \in X_{23r}, \quad r = 0, \ldots, N_{r2} \cdot N_{r3}$$
$$X_{23r} := \left\{ x \in \mathbb{R}^n \;\middle|\; \begin{array}{c} x^T M_{23r} x + N_{23r} x + C_{23r} \leq 0 \\ L_{23r} x \leq W_{23r} \end{array} \right\}$$

Se han obtenido por tanto dos particiones del espacio de estados que, nuevamente, es posible intersectar para obtener un máximo de $N_s = N_{r1} \cdot N_{r2} \cdot N_{r3}$ regiones:

$$X_s = \left\{ x \in \mathbb{R}^n \;\middle|\; \begin{array}{c} x^T M_{23r} x + N_{23r} x + C_{23r} \leq 0 \\ \left[ \begin{array}{c} L_{1i} \\ L_{23r} \end{array} \right] x \leq \left[ \begin{array}{c} W_{1i} \\ W_{23r} \end{array} \right] \end{array} \right\}$$

Para la región $X_s$ la ley de control óptima puede expresarse como:

$$\mathcal{K}_N(x) = \left\{ \begin{array}{c} G_{1i}x + h_{1i} \text{ si } V_{1N}^{OPT} < V_{23N}^{OPT} \\ G_{23r}x + h_{23r} \text{ si } V_{23N}^{OPT} < V_{1N}^{OPT} \end{array} \right. \quad \text{para } x \in X_s, \quad s = 0, \ldots, N_s$$

Al igual que para el caso $\gamma = 2$, se desea dividir de forma explícita cada región $X_s$ en dos, según cual de los dos posibles óptimos tiene un índice de coste asociado menor. El razonamiento a seguir es exactamente el mismo que el desarrollado anteriormente: las dos posibles soluciones explícitas corresponden con la proyección del óptimo sin restricciones sobre una de las caras activas de los espacios de restricciones $\widetilde{T}_1$ o $\widetilde{T}_2 \cup \widetilde{T}_3$. Nótese que, en el segundo caso, se trata de la proyección sobre una de las caras activas de $\widetilde{T}_2$ o de las de $\widetilde{T}_3$, como se ha descrito al resolver el problema para $\gamma = 2$. Por tanto, dado que todos los subconjuntos $\widetilde{T}_i$ están definidos por restricciones lineales, el lugar geométrico en el que el índice de coste obtenido proyectando sobre $\widetilde{T}_1$ es igual al obtenido proyectando sobre $\widetilde{T}_2 \cup \widetilde{T}_3$ es de nuevo una curva cuadrática de la forma (3.12). La única diferencia con el caso anterior es que las restricciones que definen la región que estamos tratando de dividir, $X_s$, ya no son todas lineales sino que pueden estar definidas a su vez por otra forma cuadrática.
Análogamente a como se operó para el caso $\gamma = 2$, se puede obtener la solución a $\mathcal{P}_N(x)$ para $\gamma = 3$ en la región $X_s$:

$$\mathcal{K}_N(x) = \left\{ \begin{array}{c} G_{1i}x + h_{1i} \text{ para } x \in X_{s1} \\ G_{23r}x + h_{23r} \text{ para } x \in X_{s2} \end{array} \right.$$

donde:

$$X_{s1} = \left\{ x \in \mathbb{R}^n \;\middle|\; \begin{array}{c} x^T[-M_{s1}]x - N_{s1}x - C_{s1} \leq 0 \\ x^T M_{23r} x + N_{23r} x + C_{23r} \leq 0 \\ \left[ \begin{array}{c} L_{1i} \\ L_{23r} \end{array} \right] x \leq \left[ \begin{array}{c} W_{1i} \\ W_{23r} \end{array} \right] \end{array} \right\}$$
$$X_{s2} = \left\{ x \in \mathbb{R}^n \;\middle|\; \begin{array}{c} x^T M_{s1} x + N_{s1} x + C_{s1} \leq 0 \\ x^T M_{23r} x + N_{23r} x + C_{23r} \leq 0 \\ \left[ \begin{array}{c} L_{1i} \\ L_{23r} \end{array} \right] x \leq \left[ \begin{array}{c} W_{1i} \\ W_{23r} \end{array} \right] \end{array} \right\}$$

De nuevo, analizando todas las posibles regiones $X_s$, se obtiene la caracterización explícita del controlador para todo el espacio de estados. En este caso,

cada región estará definida por una serie de restricciones lineales y dos restricciones cuadráticas ($\gamma - 1 = 2$).
El caso general, para $\gamma > 3$, se deduce de la observación de que siempre es posible dividir el problema de optimización en dos, uno con restricciones lineales y otro con restricciones descritas como la unión de varios espacios de restricciones lineales. Éste último a su vez puede ser dividido en otros dos problemas y, operando así sucesivamente, se llega a un último caso en que todos los subproblemas en que se ha dividido el problema original tienen restricciones lineales:

$$\mathcal{P}_N(x): \quad V_N^{OPT}(x) := \min \left\{ V_{1N}^{OPT}(x), V_{2-\gamma N}^{OPT}(x) \right\}$$

donde:

$$V_{1N}^{OPT}(x) := \min \tfrac{1}{2}\widetilde{\mathbf{u}}^T\widetilde{\mathbf{u}} + \widetilde{\mathbf{u}}^T H^{-1/2} F x,$$
sujeto a:
$$(\widetilde{\mathbf{u}}, x) \in \widetilde{T}_1$$

$$V_{2-\gamma N}^{OPT}(x) := \min \{ V_{2N}^{OPT}(x), V_{3-\gamma N}^{OPT}(x) \}$$

$$V_{2N}^{OPT}(x) := \min \tfrac{1}{2}\widetilde{\mathbf{u}}^T\widetilde{\mathbf{u}} + \widetilde{\mathbf{u}}^T H^{-1/2} F x,$$
sujeto a:
$$(\widetilde{\mathbf{u}}, x) \in \widetilde{T}_2$$

$$V_{3-\gamma N}^{OPT}(x) := \min \{ V_{3N}^{OPT}(x), V_{4-\gamma N}^{OPT}(x) \}$$

$$\vdots$$

■

### 3.4.2. Propiedades de la solución explícita

El teorema 3.2 muestra que, para el caso general, las regiones en las que se divide el espacio de estados para obtener una solución explícita al problema de optimización están delimitadas por restricciones lineales y cuadráticas. Dichas restricciones cuadráticas, para algunos casos particulares, cumplen ciertas propiedades que, como se verá en la sección 4.2, permiten simplificar la metodología seguida para obtener la caracterización explícita buscada.

Como se muestra en la prueba del teorema 3.2, las restricciones cuadráticas $Q_{ij}$ que aparecen son aquellas que dividen una determinada región lineal en función de cuál de dos hiperplanos $f_1$ y $f_2$ es el más cercano:

$$Q_{ij} := \left\{ x \in \mathbb{R}^n \,\middle|\, \mathrm{dist}(\widetilde{\mathbf{u}}_{uc}^{opt}, f_1) = \mathrm{dist}(\widetilde{\mathbf{u}}_{uc}^{opt}, f_2) \right\}$$
donde:
$$f_1 := \left\{ \widetilde{\mathbf{u}} \in \mathbb{R}^n \,\middle|\, \widetilde{\Phi}_{1l}\widetilde{\mathbf{u}} = \Theta_{1l} \right\}$$
$$f_2 := \left\{ \widetilde{\mathbf{u}} \in \mathbb{R}^n \,\middle|\, \widetilde{\Phi}_{2l}\widetilde{\mathbf{u}} = \Theta_{2l} \right\}$$

En primer lugar, se enuncia la proposición 3.3 que da una condición suficiente para que la curva $Q_{ij}$ sea un hiperplano, con la consiguiente mayor sencillez en la definición de las regiones que ello supone.

**Proposición 3.3.** *Si tanto $f_1$ como $f_2$ son hiperplanos, el subconjunto de la curva $Q_{ij}$ que cumple simultáneamente $\{\widetilde{\Phi}_{1l}\widetilde{\mathbf{u}}_{uc}^{opt}(x) \leq \Theta_{1l}\}$ y $\{\widetilde{\Phi}_{2l}\widetilde{\mathbf{u}}_{uc}^{opt}(x) \leq \Theta_{2l}\}$ es un hiperplano.* □

**Prueba.** Volviendo a la definición de $Q_{ij}$ y recordando el lema 3.1 se tiene:

$$Q_{ij} := \{\mathbf{u} \in \mathbb{R}^m \,|\mathrm{dist}(\mathbf{u}, f_1) = \mathrm{dist}(\mathbf{u}, f_2)\} = \left\{\mathbf{u} \in \mathbb{R}^m \left|\left\|\widetilde{\Phi}_{1l}^{\dagger}\left(\widetilde{\Phi}_{1l}\mathbf{u} - \Theta_{1l}\right)\right\| = \left\|\widetilde{\Phi}_{2l}^{\dagger}\left(\widetilde{\Phi}_{2l}\mathbf{u} - \Theta_{2l}\right)\right\|\right.\right\}$$

Reemplazando la expresión de la pseudoinversa:

$$\left\|\widetilde{\Phi}_{1l}^T(\widetilde{\Phi}_{1l}\widetilde{\Phi}_{1l}^T)^{-1}\left(\widetilde{\Phi}_{1l}\mathbf{u} - \Theta_{1l}\right)\right\| = \left\|\widetilde{\Phi}_{2l}^T(\widetilde{\Phi}_{2l}\widetilde{\Phi}_{2l}^T)^{-1}\left(\widetilde{\Phi}_{2l}\mathbf{u} - \Theta_{2l}\right)\right\|$$

Puesto que $\widetilde{\Phi}_{1l}$ y $\widetilde{\Phi}_{2l}$ son vectores fila, tanto $(\widetilde{\Phi}_{1l}\widetilde{\Phi}_{1l}^T)^{-1}$ como $\left(\widetilde{\Phi}_{1l}\mathbf{u} - \Theta_{1l}\right)$ son escalares. Por tanto, por la propiedad homogénea de la norma se tiene:

$$\frac{\|\widetilde{\Phi}_{1l}^T\|}{\left|(\widetilde{\Phi}_{1l}\widetilde{\Phi}_{1l}^T)\right|}\left|\left(\widetilde{\Phi}_{1l}\mathbf{u} - \Theta_{1l}\right)\right| = \frac{\|\widetilde{\Phi}_{2l}^T\|}{\left|(\widetilde{\Phi}_{2l}\widetilde{\Phi}_{2l}^T)\right|}\left|\left(\widetilde{\Phi}_{2l}\mathbf{u} - \Theta_{2l}\right)\right|$$

La solución a la igualdad anterior se corresponde con las ecuaciones de dos hiperplanos:

$$\begin{cases} \left(k_1\widetilde{\Phi}_{1l} - k_2\widetilde{\Phi}_{2l}\right)\mathbf{u} = k_1\Theta_{1l} - k_2\Theta_{2l} \\ \left(k_1\widetilde{\Phi}_{1l} + k_2\widetilde{\Phi}_{2l}\right)\mathbf{u} = k_1\Theta_{1l} + k_2\Theta_{2l} \end{cases}$$

donde:

$$k_1 = \frac{\|\widetilde{\Phi}_{1l}^T\|}{\left|(\widetilde{\Phi}_{1l}\widetilde{\Phi}_{1l}^T)\right|}$$

$$k_2 = \frac{\|\widetilde{\Phi}_{2l}^T\|}{\left|(\widetilde{\Phi}_{2l}\widetilde{\Phi}_{2l}^T)\right|}$$

Para el primero de los dos hiperplanos, es posible obtener la expresión de $\widetilde{\Phi}_{1l}\mathbf{u}$:

$$\widetilde{\Phi}_{1l}\mathbf{u} = \Theta_{1l} + \frac{k_2}{k_1}\left(\widetilde{\Phi}_{2l}\mathbf{u} - \Theta_{2l}\right)$$

Si se desea que satisfaga $\widetilde{\Phi}_{1l}\mathbf{u} \leq \Theta_{1l}$:

$$\begin{aligned} \Theta_{1l} + \frac{k_2}{k_1}\left(\widetilde{\Phi}_{2l}\mathbf{u} - \Theta_{2l}\right) &\leq \Theta_{1l} \\ \frac{k_2}{k_1}\left(\widetilde{\Phi}_{2l}\mathbf{u} - \Theta_{2l}\right) &\leq 0 \\ \widetilde{\Phi}_{2l}\mathbf{u} &\leq \Theta_{2l} \end{aligned}$$

donde se ha tenido en cuenta que $k_1$ y $k_2$ son siempre positivas. Por contra, para el segundo hiperplano:

$$\widetilde{\Phi}_{1l}\mathbf{u} = \Theta_{1l} + \frac{k_2}{k_1}\left(\Theta_{2l} - \widetilde{\Phi}_{2l}\mathbf{u}\right)$$

$$\begin{aligned}\Theta_{1l} + \frac{k_2}{k_1}\left(\Theta_{2l} - \widetilde{\Phi}_{2l}\mathbf{u}\right) &\leq \Theta_{1l}\\ \frac{k_2}{k_1}\left(\Theta_{2l} - \widetilde{\Phi}_{2l}\mathbf{u}\right) &\leq 0\\ \widetilde{\Phi}_{2l}\mathbf{u} &\geq \Theta_{2l}\end{aligned}$$

Por tanto, sólo el primer hiperplano cumple simultáneamente ambos conjuntos de restricciones. ■

### Restricciones sobre las acciones de control

Un caso particular, cuyo estudio resulta interesante, es el del problema de control predictivo con restricciones únicamente sobre las acciones de control. Las restricciones sobre las salidas o variables de estado del proceso suelen representar límites de seguridad y por tanto pueden aparecer o no en la formulación del controlador predictivo, pero las restricciones sobre las acciones de control representan los límites físicos de los actuadores, por lo que siempre están presentes.

La ausencia de restricciones sobre las salidas o las variables de estado, implica que la formulación de las matrices que engloban todas las restricciones no incluya el término función del espacio de estados:

$$\widetilde{T}_i = \{\widetilde{\mathbf{u}} \in \mathbb{R}^n \,\big|\, \widetilde{\Phi}_i\widetilde{\mathbf{u}} \leq \Delta_i\,\}$$

Es decir, para todos los espacios de restricciones $\widetilde{T}_i$, se cumple $\Lambda_i = 0$.

El problema de optimización asociado al control predictivo en este caso, sigue estando descrito por (3.3) y la solución cumple el teorema 3.2. No obstante, al cumplirse $\Lambda_i = 0$ las curvas cuadráticas que definen la partición del espacio de estados tienen una expresión más simple:

$$Q_{ij} := \left\{x \in \mathbb{R}^n \,\big|\, x^T M_{ij} x + N_{ij} x + C \geq 0\,\right\}$$

donde:

$$\begin{aligned}M_{ij} &= F^T H^{-1/2}\left(\widetilde{\Phi}_{1l}^{\dagger}\widetilde{\Phi}_{1l} - \widetilde{\Phi}_{2l}^{\dagger}\widetilde{\Phi}_{2l}\right)H^{-1/2}F\\ N_{ij} &= 2\left(\widetilde{\Phi}_{1l}^{\dagger}\Delta_{1l} - \widetilde{\Phi}_{2l}^{\dagger}\Delta_{2l}\right)^T H^{-1/2}F\\ C_{ij} &= \Delta_{1l}^T(\widetilde{\Phi}_{1l}\widetilde{\Phi}_{1l}^T)^{-1}\Delta_{1l} - \Delta_{2l}^T(\widetilde{\Phi}_{2l}\widetilde{\Phi}_{2l}^T)^{-1}\Delta_{2l}\end{aligned}$$

Para este tipo de problemas, en los que únicamente hay restricciones sobre las acciones de control, se va a mostrar como para dos casos particulares la curva $Q_{ij}$ es semidefinida positiva en uno de ellos y un hiperplano en el otro. De nuevo, como se verá en el capítulo 4 esto ofrece una mayor sencillez en la obtención el estudio y la caracterización de las regiones definidas por $Q_{ij}$.

**Definición 3.3.** Se dice que una matriz cuadrada $A$ es *idempotente* si se cumple $A \cdot A = A$ □

Las matrices idempotentes cumplen las siguientes propiedades:

- Si $A$ es idempotente, $I - A$ también lo es.
- Todos los valores propios de $A$ son 0 o 1. Por tanto, $A$ es semidefinida positiva ($A \succeq 0$)

**Proposición 3.4.** *En ausencia de restricciones sobre salidas y variables de estado ($\Lambda_i = 0$), si una curva $Q_{ij}$ está definida como aquella en que es igual la distancia a una hiperplano $f_2$ cualquiera y a otro hiperplano $f_1$ con una matriz $\widetilde{\Phi}_{1l}$ cuadrada de rango completo , la matriz $M_{ij}$ que define $Q_{ij}$ es semidefinida positiva ($M_{ij} \succeq 0$)* □

**Prueba.** Si $\widetilde{\Phi}_{1l}$ es cuadrada de rango completo, se cumple $\widetilde{\Phi}_{1l}^{\dagger} = \widetilde{\Phi}_{1l}^{-1}$ y por tanto:

$$\widetilde{\Phi}_{1l}^{\dagger}\widetilde{\Phi}_{1l} - \widetilde{\Phi}_{2l}^{\dagger}\widetilde{\Phi}_{2l} = \widetilde{\Phi}_{1l}^{-1}\widetilde{\Phi}_{1l} - \widetilde{\Phi}_{2l}^{\dagger}\widetilde{\Phi}_{2l} = I - \widetilde{\Phi}_{2l}^{\dagger}\widetilde{\Phi}_{2l}$$

Por otro lado se tiene que para cualquier matriz $A$, $A^{\dagger}A$ es una matriz idempotente:

$$\left(A^{\dagger}A\right)\left(A^{\dagger}A\right) = \left(A^T(AA^T)^{-1}A\right)\left(A^T(AA^T)^{-1}A\right) = A^T(AA^T)^{-1}A = A^{\dagger}A$$

Dado que $\widetilde{\Phi}_{2l}^{\dagger}\widetilde{\Phi}_{2l}$ es idempotente, $\widetilde{\Phi}_{1l}^{\dagger}\widetilde{\Phi}_{1l} - \widetilde{\Phi}_{2l}^{\dagger}\widetilde{\Phi}_{2l} = I - \widetilde{\Phi}_{2l}^{\dagger}\widetilde{\Phi}_{2l}$ también lo es, y por tanto $(\widetilde{\Phi}_{1l}^{\dagger}\widetilde{\Phi}_{1l} - \widetilde{\Phi}_{2l}^{\dagger}\widetilde{\Phi}_{2l}) \succeq 0$ (puesto que todos sus valores propios son 0 o 1).
Se tiene por tanto:

$$z^T(\widetilde{\Phi}_{1l}^{\dagger}\widetilde{\Phi}_{1l} - \widetilde{\Phi}_{2l}^{\dagger}\widetilde{\Phi}_{2l})z \geq 0 \qquad \forall z$$

Si tomamos $z = H^{-1/2}Fx$:

$$x^T F^T H^{-1/2}(\widetilde{\Phi}_{1l}^{\dagger}\widetilde{\Phi}_{1l} - \widetilde{\Phi}_{2l}^{\dagger}\widetilde{\Phi}_{2l})H^{-1/2}Fx \geq 0 \qquad \forall x$$

Se deduce pues que $M_{ij} \succeq 0$. ■

**Proposición 3.5.** *En ausencia de restricciones sobre salidas y variables de estado ($\Lambda_i = 0$), si una curva $Q_{ij}$ está definida como aquella en que es igual la distancia a dos hiperplanos $f_1$ y $f_2$ con las matrices $\widetilde{\Phi}_{1l}$ y $\widetilde{\Phi}_{2l}$ cuadradas de rango completo, la curva $Q_{ij}$ es a su vez un hiperplano ($M_{ij} = 0$)* □

**Prueba.** Si ambas matrices son cuadradas, tenemos:

$$\widetilde{\Phi}_{1l}^{\dagger}\widetilde{\Phi}_{1l} - \widetilde{\Phi}_{2l}^{\dagger}\widetilde{\Phi}_{2l} = \widetilde{\Phi}_{1l}^{-1}\widetilde{\Phi}_{1l} - \widetilde{\Phi}_{2l}^{-1}\widetilde{\Phi}_{2l} = I - I = 0$$

Y, por tanto, $M_{ij} = 0$ y $Q_{ij} := \{x \in \mathbb{R}^n \,|N_{ij}x + C \geq 0\}$. ■

## 3.5. Conclusiones

En este capítulo se ha planteado formalmente el problema del control predictivo con restricciones poliédricas no convexas, describiendo en primer lugar la formación de los conjuntos de restricciones para el problema definido sobre el vector de acciones de control futuras. A continuación, se han discutido las dificultades que supone plantear la resolución del problema mediante el cálculo en línea de varios problemas de programación cuadrática.

El resultado principal del capítulo es la demostración de la existencia, en analogía al caso de sistemas con restricciones poliédricas convexas, de una solución explícita al problema planteado. Dicha solución es afín a tramos definidos mediante desigualdades lineales y cuadráticas.

Por último, se han mostrado algunos casos en los que las regiones de la solución tienen una caracterización particular, que será de utilidad en el siguiente capítulo.

Capítulo 4

# Cálculo de la solución explícita

## 4.1. Introducción

En el capítulo anterior, se ha planteado formalmente el problema del control predictivo con restricciones poliédricas no convexas. Se ha demostrado que el problema de optimización que es necesario resolver (3.3) admite una solución explícita afín a tramos. Se ha visto que dichos tramos pueden definirse mediante desigualdades lineales y cuadráticas.

Aunque la prueba del teorema (3.2) es constructiva, existen motivos para desarrollar en mayor medida la obtención de la solución explícita del problema de optimización. La razón fundamental es que en la demostración del teorema se asume que, tras intersectar dos regiones de particiones solución de problemas diferentes, las dos posibles expresiones afines del óptimo son factibles (cada una a un lado de una curva cuadrática). No obstante, en el caso general esto no es cierto, puesto que en muchos casos una de las dos soluciones siempre ofrecerá un coste menor que la otra para la totalidad de la región (es decir, la curva cuadrática realmente no pasa por la región poliédrica). Es conveniente identificar estos casos por simplicidad de la solución explícita, tanto en cuanto a los requerimientos de memoria como al coste computacional.

En las siguientes secciones, se desarrolla una metodología que incorpora un procedimiento para la eliminación de las soluciones no óptimas. Además, dicha metodología utiliza de manera adecuada la información existente de las particiones solución de los subproblemas convexos para realizar la intersección de regiones de manera eficiente.

Por último, en la sección 4.4, se introduce una metodología diferente para el cálculo de la solución explícita. El procedimiento está basado en la definición de un problema simplificado cuyas restricciones son la envolvente convexa de

las restricciones originales. Dicho problema admite una solución explícita con regiones poliédricas como las vistas en la sección 2.3. A partir de esta solución, se verá como puede obtenerse la solución explícita del problema original, eliminando también las soluciones que no son óptimas en las diferentes regiones.

## 4.2. Metodología de intersección, división y unión para $\gamma = 2$

En esta sección y la siguiente se plantea un posible metodología para la obtención de la solución explícita al problema (3.3) que, como se ha visto en la sección 3.4, consiste en una solución afín a tramos definida sobre una partición cuadrática del espacio de estados.
Para obtener la solución genérica, se partirá inicialmente del caso en que las restricciones no convexas pueden dividirse en dos regiones convexas ($\gamma = 2$). En ese caso, una metodología para obtener la solución explícita podría ser el algoritmo 4.1.

**Algoritmo 4.1** Obtención de la solución explícita para $\gamma = 2$

1. Obtener la solución explícita de los problemas de optimización con restricciones convexas.
2. Intersectar las dos particiones del espacio de estados obtenidas.
3. Comprobar si es necesario dividir las regiones resultantes en función de cual de las dos posibles soluciones es la óptima.
   *a*) Si es necesario, obtener las dos subregiones.
   *b*) Si no es necesario, comprobar si la región puede unirse con alguna de las contiguas (en el caso de que tenga la misma solución afín).

A continuación, se describen los procedimientos para llevar a cabo cada uno de los pasos de dicho algoritmo.

### 4.2.1. Obtención de la solución explícita de los subproblemas

Se trata de resolver, de manera explícita, los dos problemas de programación cuadrática que nos dan el posible mínimo a nuestro problema de optimización

(3.5):

$$\mathcal{P}_N(x): \quad V_N^{OPT}(x) := \min \left\{ V_{iN}^{OPT}(x) \right\}$$

donde:

$$V_{iN}^{OPT}(x) := \min \tfrac{1}{2}\mathbf{u}^T H \mathbf{u} + \mathbf{u}^T F x,$$

sujeto a:
$\mathbf{u} \in T_i$

Es decir, se busca la solución explícita a los dos problemas de control predictivo con restricciones lineales. La solución a problemas de este tipo se describe en la sección 2.3, donde se muestra que es una función afín a tramos sobre una partición lineal del espacio de estados (2.26) y (2.27):

$$\mathcal{K}_{iN}(x) = G_{ij}x + h_{ij}, \quad \text{para } x \in X_{ij}, \quad i = 1, 2 \; j = 0, \ldots, N_i$$

donde:

$$X_{ij} := \{x \in \mathbb{R}^n \,|L_{ij}x \leq W_{ij}\,\}$$

Dependiendo de las restricciones de cada uno de los problemas, el conjunto de estados iniciales factibles, $\mathbb{S}_{iN}$, puede ser diferente. En ese caso, para algunos valores de $x$, se tendrá solución para uno de los dos problemas, pero no para el otro. Para poder realizar un tratamiento uniforme de todas las regiones se añadirán a cada solución explícita nuevas regiones sin ninguna ley de control asociada. Para ello, será necesario calcular el complemento de cada uno de los conjuntos $\mathbb{S}_{iN}$ e intersectarlo con el otro conjunto. Esto, en general, resultará en un poliedro no convexo no definido por un mínimo de elementos, por lo que será conveniente aplicar los algoritmos introducidos en la sección 2.6.2. De esta forma, el número de nuevas regiones $X_{ij}$ que es necesario añadir, será mínimo. Tras finalizar este procedimiento, la solución explícita para cada una de los dos problemas queda de la siguiente forma:

$$\mathcal{K}_{iN}(x) = \begin{cases} G_{ij}x + h_{ij}, & \text{para } x \in X_{ij}, \quad i = 1, 2; \; j = 0, \ldots, N_i \\ \emptyset, & \text{para } x \in X_{ij}, \quad i = 1, 2; \; j = N_i + 1, \ldots, N_{ri} \end{cases}$$

donde:

$$X_{ij} := \{x \in \mathbb{R}^n \,|L_{ij}x \leq W_{ij}\,\}$$

### 4.2.2. Intersección de las dos particiones del estado

Una vez obtenida la solución explícita a los dos controladores predictivos se tienen dos particiones del espacio de estados, cada una de ellas con una solución afín correspondiente a cada región. Para obtener una partición única, se deben intersectar todas las regiones poliédricas que describen ambas particiones obteniéndose:

$$X_r = X_{1i} \cap X_{2j} = \left\{ x \in \mathbb{R}^n \left| \begin{bmatrix} L_{1i} \\ L_{2j} \end{bmatrix} x \leq \begin{bmatrix} W_{1i} \\ W_{2j} \end{bmatrix} \right. \right\}, \quad r = 0 \ldots N_{r1} \cdot N_{r2}$$

No obstante, puede ocurrir que muchas de las $N_{r1} \cdot N_{r2}$ regiones así definidas estén vacías. Además, para las que no lo estén, muchas de las desigualdades que las definen pueden ser redundantes.
A continuación, se detalla cómo comprobar si las regiones fruto de la intersección son factibles y, en caso afirmativo, un procedimiento eficiente para eliminar las restricciones redundantes.

### Factibilidad de las intersecciones

Comprobar si la intersección de dos regiones no está vacía, equivale a analizar la consistencia del sistema (4.1) es factible:

$$\begin{cases} L_{1i}x - W_{1i} \leq 0 \\ L_{2j}x - W_{2j} \leq 0 \end{cases} \tag{4.1}$$

Como se vio en la sección 2.7.1 la factibilidad de sistemas lineales de este tipo puede comprobarse mediante la resolución de un LP.

De manera estricta, aunque la región factible solución al problema anterior no sea de dimensión completa, la intersección de las dos regiones existe. No obstante, a efectos prácticos en el resto del epígrafe interesa considerar que en ese caso la intersección de ambas regiones es inexistente. Por tanto, en un abuso de notación, cuando se hable de intersección de regiones de distinta partición vacía ($X_{1i} \cap X_{2j} = \emptyset$) nos estaremos refiriendo a que la intersección no tiene dimensión completa ($\dim(X_{1i} \cap X_{2j}) < n$).

### Eliminación de redundancias en las intersecciones

Una vez se ha reducido el número inicial de posibles regiones $N_{r1} \cdot N_{r2}$ a las que realmente son consistentes, $X_r$, interesa comprobar si éstas están definidas por ecuaciones redundantes, puesto que hasta el momento se han incluido todas las restricciones que definen a ambas regiones de origen.

A continuación, con objeto de eliminar las redundancias existentes en estas regiones de manera eficiente, se introducen algunas definiciones y propiedades geométricas de las dos particiones del espacio de estados.

**Definición 4.1.** Dado un poliedro $X \in \mathbb{R}^n$ definido por las desigualdades lineales $Lx \leq W$ tómese uno de los hiperplanos $h := \{x | L^i x = W^i\}$. Si se cumple que $\mathcal{F} = X \cap h$ es de dimensión $n-1$, se dice que $\mathcal{F}$ es una faceta del poliedro. □

**Definición 4.2.** Dos poliedros son contiguos si comparten una faceta común. □

**Definición 4.3.** Dos poliedros $X_1$ y $X_2$ son adyacentes si $\dim(X_1 \cap X_2) = n-1$. □

**Ejemplo 4.1.** *Supongamos que se tiene un conjunto de poliedros como el mostrado en la figura 4.1. De acuerdo a las definiciones anteriores,* $X_2$ *y* $X_3$ *serán contiguas puesto que* $X_2 \cap X_3 = \mathcal{F}_2$ *es una faceta común para ambas.*
*Sin embargo,* $X_1$ *y* $X_2$ *son adyacentes pero no contiguas puesto que a pesar de que* $\dim(X_1 \cap X_2) = n - 1 = 1$*, dicha intersección es sólo una faceta para* $X_2$ *y no para* $X_1$ *(cuya faceta para el hiperplano en cuestión es* $\mathcal{F}_1$*).* ■

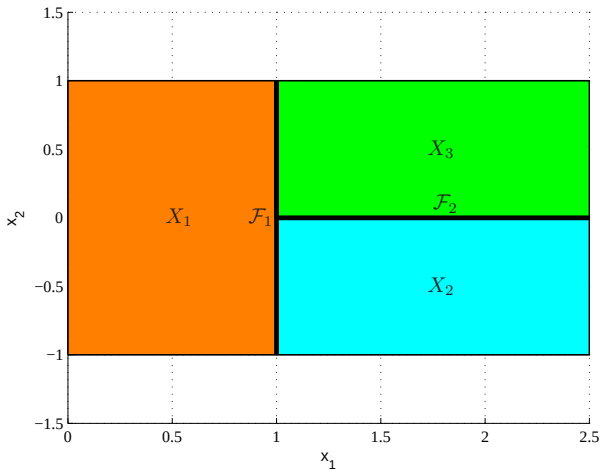


***Figura 4.1:*** Regiones contiguas y adyacentes

**Observación 4.1.** Si dos poliedros $X_1$ y $X_2$ son contiguos con una faceta común $\mathcal{F}$, $X_1$ será el único poliedro adyacente a $X_2$ a través de la faceta $\mathcal{F}$, y viceversa. □

**Observación 4.2.** Si una partición cumple la propiedad de faceta-a-faceta (definición 2.1), cualquier región tiene como máximo una región adyacente por cada faceta. □

**Teorema 4.1.** *Considérese un poliedro* $X_i := \{x \in \mathbb{R}^n \,|L_i x \leq W_i\}$ *correspondiente a la región crítica para la que la solución al problema de optimización* (3.5) *tiene activas un conjunto de restricciones* $l := \{l_1, \ldots, l_k\}$*, y tómese una de las facetas de dicho poliedro* ($\mathcal{F}$)*. Si se satisfacen las siguientes condiciones:*

- *Las filas de la matriz que define las restricciones activas,* $\Phi_l$*, son linealmente independientes.*
- *En la definición de la región (antes de eliminar las redundancias) ninguna de las desigualdades que define* $\mathcal{F}$ *aparece más de una vez.*
- *Para cualquier punto de la región, el óptimo nunca presenta restricciones activas débilmente (es decir, ningún óptimo satisface* $\lambda_i = 0$ *para* $i \in l$*).*

*la partición cumple la propiedad de faceta-a-faceta y la región contigua a* $X_i$ *a través de* $\mathcal{F}$ *puede obtenerse como:*

1. $l' = \{l_1, \ldots, l_k, l_{k+1}\}$ *si el hiperplano que define* $\mathcal{F}$ *corresponde a una desigualdad del tipo* (2.31a).

2. $l' = \{l_1, \ldots, l_{k-1}\}$ *si el hiperplano que define* $\mathcal{F}$ *corresponde a una desigualdad del tipo* (2.31b).

□

**Prueba.** Ver teorema 2 en [TJB03a]. ■

**Ejemplo 4.2.** *La figura 4.2 representa una zona de la solución explícita de un problema en la que se cumplen las condiciones del teorema 4.1. En la región* $X_1$ *no se encuentra activa ninguna de las restricciones. Una de sus regiones contiguas,* $X_2$*, presenta una restricción activa más, la de la desigualdad correspondiente a la faceta* $\mathcal{F}_1$*. En cuanto a la región* $X_3$*, tiene una restricción activa más que la* $X_2$ *por lo que es contigua a esta (tiene en común con ella la faceta* $\mathcal{F}_2$*). Como tiene dos restricciones activas más que la región* $X_1$*, no es contigua a ésta, como puede comprobarse (la intersección entre ambas no es de dimensión* $n-1$*).* ■

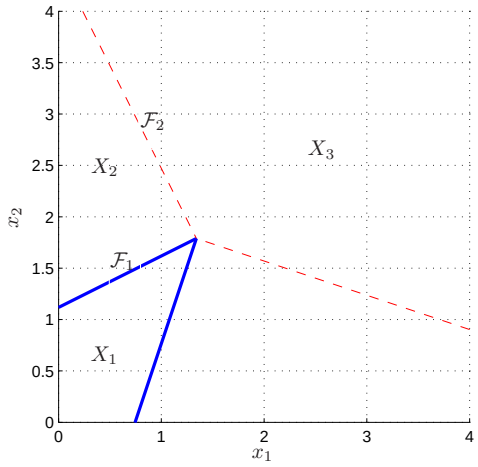


***Figura 4.2:*** Regiones contiguas en una partición

El teorema 4.1 es de utilidad para determinar qué regiones de una partición lineal son contiguas. Esto es sencillo de realizar si para obtener las regiones de dicha partición se hace uso del algoritmo propuesto en [TJB03a], puesto que en el desarrollo del propio algoritmo se hace uso de la contigüidad de las regiones, por lo que basta con almacenar dicha información.

Cuando las condiciones del teorema no se cumplen, en general cualquier región puede tener más de una región adyacente por cada faceta. Para encontrar todas esas regiones, se puede aplicar el algoritmo 2.2, descrito en la sección 2.3.

Si se tienen en cuenta las definiciones y el teorema anteriores, se puede enunciar la siguiente proposición, útil para la eliminación de ecuaciones redundantes:

**Proposición 4.2.** *Considérese una región $X_{1i}$ y el conjunto de todas las regiones de la misma partición adyacentes a $X_{1i}$ a través de una misma faceta $\mathcal{F}$:*

$$ad_{iF} = \{j \,|\, dim(X_{1j} \cap \mathcal{F}) = n-1\}$$

*y una región $X_{2i}$ correspondiente a otra partición diferente. Si se cumple:*

$$\begin{cases} X_{1i} \cap X_{2i} \neq \emptyset \\ X_{1j} \cap X_{2i} = \emptyset \quad \forall j \in ad_{iF} \end{cases}$$

*el hiperplano que define $\mathcal{F}$ será redundante en $X_{1i} \cap X_{2i}$.* □

**Prueba.** La proposición puede deducirse de una propiedad que resulta evidente de la definición de regiones adyacentes: no hay ningún punto de $\mathcal{F}$ que no pertenezca a alguna de las regiones $X_{1j}$ para $j \in ad_{iF}$. Por tanto, si se tiene que $X_{1j} \cap X_{2i} = \emptyset$ para todas las $j \in ad_{iF}$, $\mathcal{F} \cap X_{2i} = \emptyset$, y por tanto el hiperplano que define $\mathcal{F}$ es redundante en $X_{1i}$. ■

**Ejemplo 4.3.** *Se tiene una región $X_{12}$ con dos regiones de su misma partición adyacentes a través de la faceta $\mathcal{F}$, $X_{11}$ y $X_{13}$ (figura 4.3). Por otro lado, se tiene una región correspondiente a otra partición, $X_{21}$, que cumple $X_{11} \cap X_{21} = \emptyset$ y $X_{13} \cap X_{21} = \emptyset$ . Por tanto, como se observa en la figura, la faceta $\mathcal{F}$ es redundante en $X_{12} \cap X_{21}$.* ■

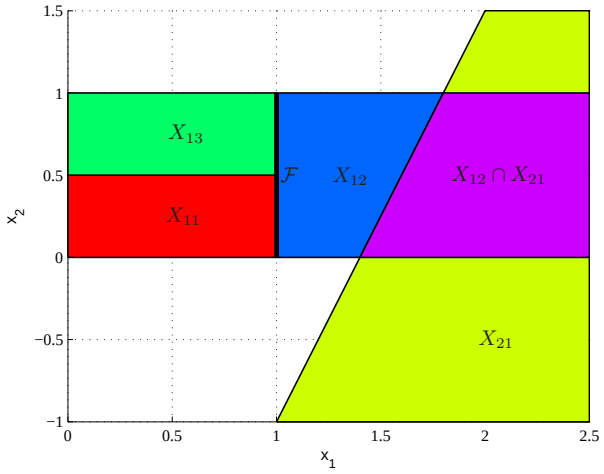


***Figura 4.3:*** Redundancia de restricción en regiones adyacentes

La proposición 4.2 es de utilidad para la eliminación de algunas de las ecuaciones redundantes en las regiones de intersección si se ha comprobado previamente la consistencia de todas las posibles intersecciones de regiones de las dos

particiones, y conociendo cuáles son las regiones adyacentes a cada región de las dos particiones, haciendo uso del algoritmo 2.2. De esta forma, sabemos qué parejas de regiones $(X_{1i}, X_{2j})$ tienen intersección vacía y cuales no. Haciendo uso de la proposición y buscando regiones adyacentes, es posible eliminar algunos de los hiperplanos redundantes.
No obstante, con este procedimiento todavía pueden quedar ecuaciones redundantes que no se hayan eliminado: aquellos hiperplanos correspondientes a facetas de una de las regiones intersectadas a través de las cuales hay regiones adyacentes que tienen intersección no vacía con la región de la segunda partición intersectada. En este tipo de casos, como se aprecia en el ejemplo 4.4, a pesar de tener información a priori de la consistencia de regiones no puede decirse nada respecto a la redundancia de alguna de sus facetas.

**Ejemplo 4.4.** *Se tiene una región $X_{12}$ con dos regiones de su misma partición adyacentes a través de una faceta $\mathcal{F}$ ($X_{11}$ y $X_{13}$) y una región de otra partición $X_{21}$ con intersección no nula con las tres regiones anteriores. Dependiendo de la disposición de $X_{21}$ la faceta $\mathcal{F}$ puede ser redundante en $X_{12} \cap X_{21}$ o no, por lo que no puede sacarse ninguna conclusión con la información disponible (figura 4.4).* ■

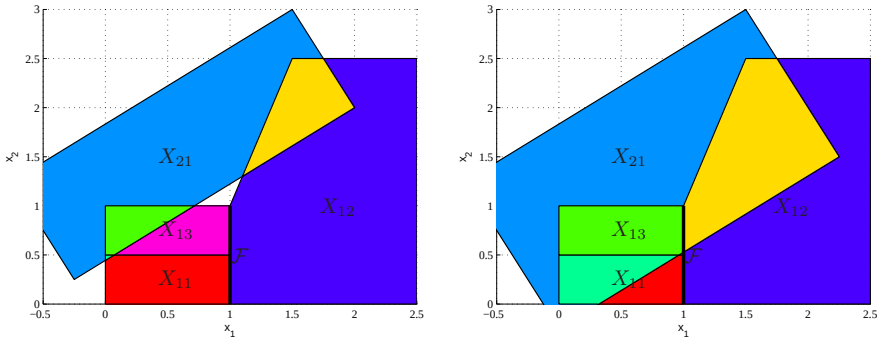


***Figura 4.4:*** Indefinición de redundancia.

Por lo tanto, para comprobar si una faceta es redundante en una determinada intersección cuando no se cumplen las premisas de la proposición 4.2, es necesario resolver un nuevo problema de factibilidad:

$$\begin{cases} x \in X_{1i} \\ x \in X_{2i} \\ x \in h \end{cases} \tag{4.2}$$

donde

$$X_{1i} := \{x \in \mathbb{R}^n \,|L_{1i}x \leq W_{1i}\}$$
$$X_{2i} := \{x \in \mathbb{R}^n \,|L_{2i}x \leq W_{2i}\}$$
$$\mathcal{F} := X_{1i} \cap h$$
$$h := \left\{x \in \mathbb{R}^n \,\middle|\, \left[L_h\right]_{[1\times n]} x = \left[W_h\right]_{[1\times n]}\right\}$$

El problema anterior es de nuevo un problema con restricciones lineales, que puede resolverse mediante un problema LP, como se vio en la sección 2.7.1. En este caso, si el sistema es factible, hay algún punto del hiperplano $h$ que se encuentra en la intersección de las dos regiones, y por lo tanto no es redundante.

Aunque la resolución del problema de factibilidad anterior sirve para todos los casos en que no es de aplicación la proposición 4.2, con objeto de reducir el número de optimizaciones que es necesario resolver para la eliminación de redundancias es interesante formular la siguiente proposición:

**Proposición 4.3.** *Dadas dos regiones de una partición del espacio de estados contiguas, $X_{1i}$ y $X_{1j}$, con una faceta común $\mathcal{F}$, y una región de otra partición diferente, $X_{2i}$, si $X_{1i} \cap X_{2i} \neq \emptyset$ y $X_{1j} \cap X_{2i} \neq \emptyset$, $\mathcal{F}$ es redundante en $X_{1i} \cap X_{2i}$ y en $X_{1j} \cap X_{2i}$ si y sólo si dicha faceta no se encuentra dentro de la región $X_{2i}$.* □

**Prueba.** Si dicha faceta se encontrara dentro de $X_{2i}$, estaría también en $X_{1i} \cap X_{2i}$ y en $X_{1j} \cap X_{2i}$, y por tanto no sería redundante en ellas. Por otro lado, si se encuentra fuera, no puede ser uno de los hiperplanos que definen la región de la intersección, por lo que será redundante. ■

**Ejemplo 4.5.** *Se tienen dos regiones contiguas de una partición del espacio de estados, $X_{11}$ y $X_{12}$, separadas por la faceta $\mathcal{F}_1$ (figura 4.5). Por otro lado, se tiene una región correspondiente a otra partición, $X_{21}$, que cumple $X_{11} \cap X_{21} \neq \emptyset$ y $X_{12} \cap X_{21} \neq \emptyset$. Como $\mathcal{F}_1$ se encuentra fuera de la región $X_{21}$, es redundante tanto en $X_{11} \cap X_{21}$ como en $X_{12} \cap X_{21}$.* ■

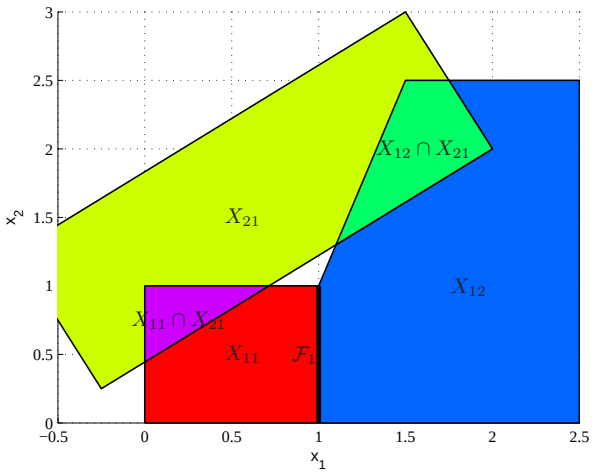


***Figura 4.5:*** Redundancia de restricción en regiones contiguas

La proposición 4.3 supone un caso particular en el que, al compartir dos regiones una misma faceta, no es necesario resolver (4.2) para cada una de ellas,

sino que con la resolución de un sólo problema de programación lineal puede eliminarse una faceta redundante en ambas regiones. En el caso general de regiones adyacentes pero no contiguas, esto no puede hacerse porque aunque las regiones compartan un mismo hiperplano, la faceta que las define es diferente.

Cuando se cumplen las condiciones de dicha proposición, para comprobar la redundancia de la faceta común a las dos regiones contiguas, $\mathcal{F}$, basta con saber cuando esta faceta se encuentra fuera de la región $X_{2i}$, es decir comprobar la factibilidad de:

$$\begin{cases} x \in X_{2i} \\ x \in \mathcal{F} \end{cases} \tag{4.3}$$

La faceta común $\mathcal{F}$ puede definirse de diferentes maneras:

$$\mathcal{F} := X_{1i} \cap h = X_{1j} \cap h = X_{1i} \cap X_{1j}$$

donde $X_{1j}$ es la región contigua a $X_{1i}$ a través de $\mathcal{F}$:

$$X_{1j} := \{x \in \mathbb{R}^n \,|L_{1j}x \leq W_{1j}\}$$

Atendiendo a las tres diferentes formas de definir la faceta, el problema de factibilidad puede resolverse mediante tres LPs diferentes (dos de ellos con restricciones de igualdad y desigualdad y un tercero con únicamente restricciones de desigualdad). Resolver un problema o el otro puede suponer pequeñas diferencias en términos de coste computacional, dependiendo del número de restricciones que definan cada una de las regiones.

Para cualquiera de ellos, si el sistema es inconsistente $\mathcal{F}$ no se encuentra en $X_{2i}$. Por lo tanto, el hiperplano $h$ es redundante tanto en $X_{1i} \cap X_{2i}$ como en $X_{1j} \cap X_{2i}$.

Haciendo uso de las proposiciones enumeradas es posible establecer un algoritmo que, a partir de dos particiones lineales del espacio de estados, obtenga de manera eficiente una única partición del espacio en la que las regiones estén definidas sin redundancias (algoritmo 4.2).

### 4.2.3. División de la regiones resultantes

Tras los pasos anteriores, se dispone de una partición del espacio de estados con regiones lineales sin redundancias, con una o dos posibles soluciones al problema de optimización, cada una de ellas con un índice de coste asociado (3.11). Para las regiones en las que existen dos posibles soluciones, la ley de control puede expresarse como:

$$\mathcal{K}_N(x) = \begin{cases} G_{1i}x + h_{1i} \text{ si } V_{1N}^{OPT}(x) < V_{2N}^{OPT}(x) & \text{para } x \in X_r, \\ G_{2j}x + h_{2j} \text{ si } V_{2N}^{OPT}(x) < V_{1N}^{OPT}(x) & r = 0, \ldots, N_r \end{cases} \tag{4.4}$$

donde $X_r := \{x \in \mathbb{R}^n \,|L_r x \leq W_r\}$ son las regiones con dos soluciones diferentes que se obtienen de intersectar las dos particiones del estado.

**Algoritmo 4.2** Intersección de dos particiones del espacio de estados.

1. Para cada región de la primer partición, $X_{1i}$:
   - Para cada faceta, comprobar condiciones del teorema 4.1. Si se cumplen, aplicar teorema para buscar la región contigua. Si no, aplicar algoritmo 2.2 para encontrar regiones adyacentes.
2. Buscar de igual forma regiones contiguas y adyacentes de la segunda partición.
3. Para cada región de la primera partición, $X_{1i}$, determinar qué regiones de la segunda partición, $X_{2j}$, tienen una intersección (4.1) factible.
4. Eliminación de redundancias para cada intersección ($X_{1i} \cap X_{2j} \neq \emptyset$):
   *a*) Para cada faceta $\mathcal{F}$ de $X_{1i}$ comprobar si hay una región contigua o varias adyacentes.
   - Si hay una región contigua, $X_{1k}$, comprobar $X_{1k} \cap X_{2j}$:
     - Si ($X_{1k} \cap X_{2j} = \emptyset$) , $\mathcal{F}$ es redundante en ($X_{1i} \cap X_{2j}$) (proposición 4.2).
     - Si ($X_{1k} \cap X_{2j} \neq \emptyset$) comprobar si $\mathcal{F}$ se encuentra en el interior de $X_{2j}$ (problema de factibilidad (4.3)):
       - Si no es factible, $\mathcal{F}$ es redundante en ($X_{1i} \cap X_{2j}$) y en ($X_{1k} \cap X_{2j}$) (proposición 4.3).
   - Si hay varias regiones adyacentes $\{X_{1k}, X_{1l}, \ldots\}$, comprobar ($X_{1k} \cap X_{2j}$),($X_{1l} \cap X_{2j}$),etc...:
     - Si todas las restricciones están vacías, $\mathcal{F}$ es redundante en ($X_{1i} \cap X_{2j}$) (proposición 4.2).
     - Si alguna no está vacía, comprobar la redundancia de $\mathcal{F}$ resolviendo (4.2).

   *b*) Para cada faceta $\mathcal{F}$ de $X_{2j}$, proceder análogamente.

Una posible solución al problema sería implementar el algoritmo en línea del control predictivo explícito con las regiones lineales de forma que para cada valor de $x$, se determine en qué región $X_r$ se encuentra el sistema y se comparen los índices de coste $V_{iN}^{OPT}(x)$ que producen las dos posibles soluciones afines, aplicando aquella que produce el menor. No obstante, esta solución presenta dos problemas que incrementan el coste del algoritmo en línea:

- A medida que crece el número de regiones $\gamma$ en las que se divide el espacio no convexo, el coste computacional de calcular y comparar los índices de coste crece de manera lineal.
- En la división lineal de espacio de estados de la que se dispone, puede haber regiones que no sólo tengan la misma solución afín en todo su

interior, sino que además tengan la misma expresión de dicha solución que alguna de sus regiones contiguas, con lo que sería posible unirlas, reduciendo significativamente el número de regiones finales, y por tanto el coste de decidir en cual de ellas se encuentra el estado en un determinado instante.

Por tanto, resulta interesante estudiar en el algoritmo fuera de línea si cada una de las regiones lineales está dividida en subregiones con diferente solución afín, o por el contrario la expresión de dicha solución es la misma para toda la región.

En la prueba del teorema 3.2 se vio que el lugar geométrico de puntos del espacio de estado en que dos expresiones afines producen un mismo índice de coste está descrito por la ecuación de una forma cuadrática (3.12). Dicha expresión obtiene la curva cuadrática a partir de las matrices que definen las caras activas correspondientes a las dos soluciones explícitas con las que se está trabajando.

No obstante, puede ocurrir que no se disponga explícitamente de dichas matrices, porque las soluciones explícitas de cada uno de los dos subproblemas se hayan obtenido, en lugar de con el enfoque geométrico, con cualquier otro de los procedimientos descritos en la sección 2.3. En ese caso, es necesario obtener la curva a partir de unas soluciones explícitas dadas en la forma genérica de (4.4).
Para ello, basta con obtener la diferencia entre los índices cuadráticos que se obtienen aplicando cada una de las dos posibles soluciones afines:

$$\begin{aligned} Q_{ij}(x) = \quad & V_{1N}^{OPT}(x) - V_{2N}^{OPT}(x) = \tfrac{1}{2}\left((\mathbf{u}_1^{opt})^T H \mathbf{u}_1^{opt} - (\mathbf{u}_2^{opt})^T H \mathbf{u}_2^{opt}\right) + \\ & + \tfrac{1}{2}\left((\mathbf{u}_1^{opt} - \mathbf{u}_2^{opt})^T F x\right) = x^T M_{ij} x + N_{ij} x + C_{ij} \end{aligned} \tag{4.5}$$

donde

$$\begin{aligned} M_{ij} &= G_{1i}^T H G_{1i} - G_{2i}^T H G_{2i} + G_{1i}^T F - G_{2i}^T F \\ N_{ij} &= 2\left(h_{1i}^T H G_{1i} - h_{2i}^T H G_{2i} + h_{1i}^T F - h_{2i}^T F\right) \\ C_{ij} &= h_{1i}^T H h_{1i} - h_{2i}^T H h_{2i} \end{aligned}$$

En el caso general, la curva cuadrática no tiene porqué pasar por la intersección de las regiones cuyas soluciones afines la definen. Por tanto, en este punto es necesario saber si el sistema de ecuaciones (4.6) tiene alguna posible solución, es decir, si es consistente.

$$\begin{cases} L_r x \leq W_r \\ x^T M_{ij} x + N_{ij} x + C_{ij} = 0 \end{cases} \tag{4.6}$$

En el caso de que sí sea consistente, es evidente que un subconjunto de los puntos que satisfacen $L_r x \leq W_r$ cumple $x^T M_{ij} x + N_{ij} x + C_{ij} > 0$ mientras que para el resto $x^T M_{ij} x + N_{ij} x + C_{ij} < 0$. Para cada uno de ambos subconjuntos, la solución afín que produce un menor índice de coste es diferente, por lo tanto debemos dividir la región lineal en dos.

En el caso contrario, si (4.6) es inconsistente, para toda la región lineal la solución afín que produce un menor índice de coste es la misma, y no será necesario dividirla.

### Casos particulares

Antes de resolver el problema (4.6) para un caso general, se van a analizar los casos particulares que se derivan cuando se cumplen las proposiciones 3.3,3.4 y 3.5. Es importante tratar estos casos de manera particular porque el coste computacional de la resolución del problema va a ser mucho menor:

- **Caras activas de dimensión** $[1 \times m]$**:** En este caso es aplicable la proposición 3.3 y, por lo tanto, la curva cuadrática es en realidad un hiperplano ($M_{ij} = 0$). El problema (4.6) puede resolverse como un LP de la siguiente forma:

$$\begin{aligned} \min \quad & s \\ \text{sujeto a:} \quad & L_r x - W_r \leq \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} s \\ & N_{ij} x + C_{ij} = 0 \end{aligned} \tag{4.7}$$

  Si el valor del óptimo obtenido es $s > 0$, el problema es no factible y, por lo tanto, $N_{ij}x + C_{ij} = 0$ no pasa por la región formada por las restricciones lineales.

- **Restricciones sobre acciones de control y una de las caras activas de dimensión** $[m \times m]$**:** En este caso es aplicable la proposición 3.4 y, por lo tanto, la función que define la curva cuadrática, $f(x) = x^T M_{ij} x + N_{ij} x + C_{ij}$ es convexa, puesto que se cumple $M_{ij} \succeq 0$.
  En este caso, para resolver (4.6) se representa el poliedro $L_r x \leq W_r$ mediante sus vertices, $v_i$. Así cualquier punto $x$ que se encuentre en el poliedro, se puede calcular como:

$$\begin{aligned} x &= \sum_i \lambda_i v_i \\ \sum_i \lambda_i &= 1 \\ \lambda_i &\geq 0 \end{aligned}$$

  A continuación, se calcula el valor de la función $f(x)$ en cada uno de dichos vértices y dependiendo del signo que tome, se procede de diferentes formas:

  - Si hay vértices con diferentes signo, $f(v_i) > 0$ y $f(v_j) < 0$, al ser $f(x)$ una función continua, existe algún $x_0$ perteneciente al poliedro para el que $f(x_0) = 0$, por lo que el sistema (4.6) es consistente.

- Si todos los vértices cumplen $f(v_i) < 0$, el sistema es inconsistente, puesto que por convexidad $f(x) \leq \sum_i \lambda_i f(v_i)$ y, al ser $\lambda_i \geq 0$, se tiene $f(x) \leq 0$.
- Por último, si todos los vértices cumplen $f(v_i) > 0$, es necesario buscar el mínimo dentro de la región lineal resolviendo el QP:

$$\begin{array}{ll} \min & f(x) \\ \text{sujeto a:} & L_r x \leq W_r \end{array}$$

Si en este caso el valor del mínimo es $f(x^{opt}) > 0$, el sistema es inconsistente (puesto que no hay ningún $x$ que cumpla $f(x) = 0$). Si $f(x^{opt}) < 0$, el sistema será consistente, por argumentos análogos a los expuestos en el primer caso.

- **Restricciones sobre acciones de control y las dos caras activas de dimensión** $[m \times m]$**:** Se trata de un caso particular del anterior en el que se tiene $M_{ij} = 0$ (proposición 3.5), por lo que de nuevo el problema puede formularse como un LP de la forma (4.7).

### Caso general

Para el caso general la matriz $M_{ij}$ es indefinida, y por tanto para determinar la factibilidad del sistema (4.6) no es posible aplicar ninguna de las simplificaciones anteriormente descritas. No obstante, en algunos casos todavía es posible decidir de una manera relativamente sencilla si una de las dos soluciones posibles es mejor que la otra para toda la región. Esto es posible si se tiene en cuenta que la curva cuadrática $Q_{ij}(x)$ está definida como la diferencia de dos funciones cuadráticas convexas (4.5).

Para ello, en primer lugar es necesario obtener el máximo y el mínimo de ambas funciones cuadráticas ($V_{1N}^{OPT}(x)$ y $V_{2N}^{OPT}(x)$) en la región lineal. Al tratarse de curvas semidefinidas positivas, el mínimo puede obtenerse mediante la resolución de un problema de programación cuadrática estándar:

$$\begin{array}{ll} \min & V_{iN}^{OPT}(x) \\ \text{sujeto a:} & L_r x \leq W_r \end{array}$$

En cuanto al máximo, al tratarse de funciones convexas, puede demostrarse que éste se encuentra en un vértice de la región [BSS06] y por tanto puede obtenerse evaluando la función en cada uno de dichos vértices:

$$\max(V_{iN}^{OPT}) = \max_{v_j}\{V_{iN}^{OPT}(v_j)\}$$

donde $v_j$ son los vértices del poliedro $X_r$. De esta forma, si se tiene $\max(V_{1N}^{OPT}) \leq \min(V_{2N}^{OPT})$ evidentemente se cumple que $V_{1N}^{OPT}(x) < V_{2N}^{OPT}(x)$ para cualquier punto de $X_r$ y por tanto, la segunda solución puede descartarse. De manera recíproca, podría descartarse la primera solución.

Cuando la condición anterior no se cumple, no es posible afirmar que es necesario mantener ambas soluciones, puesto que todavía es posible que una de ellas tenga un coste menor que la otra para todos los puntos de la región. No obstante, el procedimiento es de utilidad, especialmente en regiones muy pequeñas.

Si el procedimiento anterior no es capaz de descartar una de las soluciones, y teniendo en cuenta que $M_{ij}$ es indefinida, no basta con resolver un problema de programación cuadrática (o lineal). En su lugar, se empleará el método introducido en la sección 2.7.2 para la resolución general del problema de factibilidad (4.8) con igualdades y desigualdades polinómicas:

$$\begin{cases} f_i(x) = 0, & (i = 1, \ldots, m) \\ g_i(x) \geq 0, & (i = 1, \ldots, p) \end{cases} \tag{4.8}$$

En dicha sección, se enuncia el teorema 2.5 que proporciona alternativas excluyentes para la factibilidad de (4.8). Dicho teorema se basa en la búsqueda de unos polinomios $F(x)$ (perteneciente al ideal asociado a las ecuaciones $f_i$) y $G(x)$ (perteneciente al cono asociado a las desigualdades $g_i$) que cumplan $F(x) + G(x) = -1$.

En la sección 2.7.2 se vio que para garantizar la inconsistencia de (4.8) basta con encontrar unos polinomios $F(x)$ y $G(x)$ concretos. Para ello, se introdujo la metodología de programación de suma de cuadrados de [Par00] que, siempre que se fije el orden de los polinomios, muestra cómo es posible formular el problema en términos de programación semidefinida (SDP).

En sentido contrario, y aunque el teorema 2.5 enuncia condiciones excluyentes, demostrar la consistencia del sistema es mucho más complicado en la práctica. Esto es debido a que, para un caso general, no existen cotas para el orden de los polinomios $F(x)$ y $G(x)$. Es decir, sería necesario comprobar que no existe ninguna pareja de polinomios (de ningún orden) que cumplan $F(x) + G(x) = -1$. Garantizar esto implicaría resolver problemas SDP de tamaño infinito.

No obstante, si se tiene presente el problema real (4.6) para el que se está comprobando la consistencia del sistema, queda claro que un condición suficiente para la inconsistencia puede bastar: el problema a resolver consiste en comprobar si una curva cuadrática pasa por una región definida por desigualdades lineales. Si al resolver (2.49) se obtiene solución, la curva cuadrática no cruza la región, y por lo tanto no será necesario dividir ésta, puesto que la expresión afín del óptimo solución del problema será la misma para toda ella. Por otro lado, si para el orden escogido para los polinomios no se encuentra solución a (2.49), se supondrá que la curva pasa por la región lineal, por lo que habrá que dividir ésta en dos con la curva cuadrática. En el caso de que realmente existieran polinomios $F(x)$ y $G(x)$ que cumplan $F(x)+G(x) = -1$, pero sean de un orden superior al supuesto al resolver el SDP, se estará dividiendo una región lineal sin necesidad de hacerlo, pero el óptimo que se obtendrá en la implementación en línea del algoritmo será el correcto (puesto que cualquier $x$ que se encuentre dentro de la región lineal estará siempre al mismo lado de

la curva cuadrática).

No obstante, aunque no exista garantía de ello, en la práctica si (4.8) es inconsistente se suelen encontrar polinomios de orden bajo que lo refutan, por lo que la problemática anteriormente descrita es poco habitual.

Tras resolver el problema (2.49) es posible decidir qué regiones $X_r$ es necesario dividir porque tiene dos soluciones afines diferentes, y cuales no. Para estas últimas, todavía es necesario averiguar a qué lado de la curva cuadrática se encuentran, y por tanto qué solución afín les corresponde. Como se acaba de comprobar que la región no está dividida por la curva (4.5) correspondiente, basta con comprobar a qué lado se encuentra un único punto factible de la región. En particular, se puede utilizar el punto $x_f$ que se obtiene al resolver el problema de programación lineal cuando se está comprobando la factibilidad de la intersección de regiones.

El algoritmo 4.3 resume el procedimiento para llevar a cabo las divisiones de las regiones lineales mediante curvas cuadráticas.

**Algoritmo 4.3** División de regiones lineales

1. Para cada región $X_r$ obtener mediante (3.12) o (4.5) los índices de coste de las dos posibles soluciones afines y la curva cuadrática en la que se igualan.
2. Calcular los máximos y mínimos de los dos índices de coste en la región, $\max(V_{iN}^{OPT})$, $\min(V_{iN}^{OPT})$.
3. Si $\max\,(V_{iN}^{OPT}) \leq \min\,(V_{jN}^{OPT})$, descartar la solución $j$.
4. Si no, resolver el problema de programación SOS (2.49) siendo la única $f_i(x)$ la curva (3.12) y las $g_i(x)$ las desigualdades $L_r x \leq W_r$
   *a*) Si el optimizador encuentra solución a (2.49), (2.46) es incompatible y por tanto la curva no pasa por la región y no es necesario dividirla. La solución afín correspondiente a $X_r$ será única:
   $$\mathcal{K}_{Nr}(x) = G_r x + h_r \tag{4.9}$$
   donde $G_r = G_{1i}$ y $h_r = h_{1i}$ si para un punto $x_f$ cualquiera de la región $x_f^T M_{ij} x_f + N_{ij} x_f + C_{ij} \leq 0$ y $G_r = G_{2i}$ y $h_r = h_{2i}$ si $x_f^T M_{ij} x_f + N_{ij} x_f + C_{ij} > 0$.
   *b*) Si no la encuentra, la región $X_r$ se divide en dos regiones por la curva (4.5), cada una de ellas con una solución afín diferente:
   $$\mathcal{K}_{Nr}(x) = \begin{cases} G_{1i}x + h_{1i} \text{ si } x^T M_{ij} x + N_{ij} x + C_{ij} \leq 0 \\ G_{2j}x + h_{2j} \text{ si } x^T M_{ij} x + N_{ij} x + C_{ij} > 0 \end{cases} \tag{4.10}$$

### 4.2.4. Minimización del número de regiones finales

Tras ejecutar el algoritmo anterior, se dispone de una partición del estado en la que las regiones que la forman, $X_r$, han sido obtenidas mediante la intersección de las regiones correspondientes a dos particiones diferentes: $X_r = X_{1i} \cap X_{2j}$. Con el procedimiento seguido, se ha supuesto que cada una de dichas regiones tiene una solución afín diferente, e incluso dos soluciones diferentes en una misma región, divididas por una curva cuadrática.
No obstante, es posible que a diferentes regiones así formadas les corresponda la misma solución. Esto es así debido a que, como se ha visto, la solución de una determinada región $X_1 = X_{1i} \cap X_{2j}$, en el caso en que no haya sido necesario dividirla, es o bien la correspondiente a $X_{1i}$ o la correspondiente a $X_{2j}$. Si se toma otra región $X_2 = X_{1i} \cap X_{2k}$, en la que una de las regiones de las que es intersección sea la misma que para $X_1$, resulta obvio que la solución puede ser la misma (si en ambos casos es la correspondiente a $X_{1i}$), por lo que en principio dichas regiones podrían unirse.
Cabe destacar que es muy importante unir dichas regiones cuando sea posible, puesto que la reducción de regiones de la partición final reduce el coste computacional del algoritmo en línea.

Por tanto, dado un poliedro no convexo formado por un conjunto $X_r$ de poliedros con la misma solución afín, es conveniente sustituirlo por un nuevo poliedro no convexo definido por la unión del mínimo número de poliedros $X_s$ posible.

Para ello, se aplicarán los algoritmos de minimización del número de elementos de un poliedro no convexo introducidos en la sección 2.6.2.
Para la aplicación de éstos, se vio que es necesario obtener, por un lado, la disposición de hiperplanos $A = \{H_i\}_{i=1,\dots,n}$ que definen todas las regiones $X_r$ y por otro las marcas de los diferentes poliedros. Para ello, se definió el vector de signos simplificado como:

$$SV_i(x) = \begin{cases} - & \text{si } a_i^T x \leq b_i \\ + & \text{si } a_i^T x \geq b_i \end{cases} \quad \text{para } i = \{1 \dots n\}$$

A partir de dicha definición, es posible expresar cualquier poliedro como una *celda* de la disposición de planos:

$$P_m = \{x \in \mathbb{R}^d | SV(x) = m\}$$

donde el vector $m$ es la *marca* del poliedro $P_m$ en la disposición de hiperplanos $A$.

La disposición de hiperplanos $A$ y las marcas $m$ pueden obtenerse en un caso general mediante algoritmos propuestos en [GTM08], así como mediante otros procedimientos como la búsqueda inversa, descrita en [AF96] y [FFL01]. No obstante, todas estas propuestas requieren la resolución de un determinado número de problemas LP.
A continuación se describe un procedimiento que, a partir de la información disponible del problema obtenida en las secciones anteriores, calcula la disposición

de hiperplanos y las marcas de los poliedros sin necesidad de resolver ningún nuevo problema de optimización. Para ello, se parte de la siguiente proposición:

**Proposición 4.4.** *Dadas dos particiones del espacio de estado en las que las soluciones afines correspondientes a cada una de las regiones $X_{1i}$ y $X_{2j}$ sean diferentes, las únicas regiones de la partición $X_r$, obtenida como intersección de las dos anteriores, cuya solución afín puede ser la misma son aquellas formadas mediante la intersección de una misma región de una de las particiones de origen.* □

**Prueba.** Como acabamos de ver en la sección 4.2.3, la función afín a tramos solución de nuestro problema de optimización en cualquier región de la partición de intersección ($X_r = X_{1i} \cap X_{2j}$) es o bien una de las soluciones afines de las regiones cuya intersección las forma ($X_{1i}$ o $X_{2j}$), o bien ambas, divididas por una curva cuadrática. Dado que suponemos que las soluciones de las regiones de las particiones de origen son diferentes, la única manera de que dos regiones $X_r$ tengan una misma solución es que estén formadas como intersección de una misma región, $X_{1i}$ o $X_{2j}$, con otra región diferente, y que la curva cuadrática correspondiente no la divida. ■

Para diseñar el procedimiento para la unión de regiones, teniendo presente la proposición 4.4, se partirá de una región de una de las particiones de origen, $X_{1i}$, y se buscarán todas las regiones $X_r$ formadas como intersección de dicha región con alguna de las regiones de la otra partición $\{X_r | X_r = X_{1i} \cap X_{2j}, j = 1, \dots N_{r2}\}$.
Evidentemente, algunas de las regiones tendrán la misma solución afín (la correspondiente a $X_{1i}$), pero otras no. Además, la unión de todas las regiones $X_r$ seleccionadas de esta manera forman un poliedro convexo, la región $X_{1i}$. Por tanto, es posible aplicar los algoritmos de unión de regiones descritos.
Para formar la disposición de hiperplanos $A$, se enumeran todas las desigualdades que definen cada una de las regiones $X_r$ consideradas, y se eliminan todas aquellas desigualdades $\sigma A_i x \leq \sigma \alpha_i$, donde $A_i x \leq \alpha_i$ es una desigualdad ya considerada. De esta manera se eliminar por un lado las desigualdades duplicadas (con $\sigma > 0$) y por otro uno de los hiperplanos que delimitan dos regiones adyacentes, una por cada una de sus caras (con $\sigma < 0$).

$$A = \{H_i : A_i x \leq \alpha_i \quad , i = 1 \dots n\} \tag{4.11}$$

En cuanto a la obtención de las marcas de las regiones $X_r$, ésta es sencilla puesto que tenemos un punto factible de cada una de dichas regiones, $x_{fr}$, obtenido tras resolver el problema de factibilidad. Dado que, por definición, todos los puntos de cada región tienen la misma marca (puesto que si hubiera puntos con distinta marca la región estaría dividida por otro hiperplano), basta con comprobar el signo que toman cada una de las desigualdades de la disposición de hiperplanos para el punto conocido de cada región:

$$m_r = SV(x_{fr}) \tag{4.12}$$

Una vez obtenidas la disposición de planos y las marcas de cada región, es posible aplicar alguno de los algoritmos de [GTM08] para realizar la unión de regiones.

El algoritmo 4.4 describe el procedimiento que une todas las regiones posibles, minimizando el número de regiones finales obtenidas.

**Algoritmo 4.4** Minimización del número de regiones lineales

1. Para cada región $X_{1i}$ buscar las regiones $\{X_r | X_r = X_{1i} \cap X_{2j}, j = 1, \dots N_{r2}\}$.
2. Comprobar la solución afín de cada $X_r$.
   - Si todas tienen la misma solución, quedarse únicamente con $X_{1i}$.
   - Si ninguna tiene la misma solución, pasar a la siguiente región $X_{1i}$.
   - Si sólo algunas tienen la misma solución, continuar con el algoritmo.
3. Obtener la disposición de planos $A$ como (4.11), eliminando desigualdades innecesarias.
4. Obtener marcas de todas las $X_r$ según (4.12).
5. Aplicar algoritmo de unión de regiones:
   - Si no se desea solapamiento, aplicar algoritmo de ramificación y poda de [GTM08].
   - Si se permite solapamiento, aplicar algoritmo de minimización de lógica booleana de [GTM08].
6. Repetir procedimiento para cada región $X_{2j}$.

Evidentemente, la metodología descrita es sólo válida si se cumple la proposición 4.4 y por tanto no incluye la posibilidad de unir regiones $X_r$ formadas mediante la intersección de regiones diferentes de ambas particiones del espacio de estado, pero cuya solución afín sea igual porque lo es para regiones diferentes de las particiones de origen. Estos casos podrán en general evitarse con una adecuada selección de las regiones convexas de partida y, en caso contrario, puede generalizarse el algoritmo descrito de una manera relativamente sencilla.

**Ejemplo 4.6.** *Se desea obtener un controlador explícito para el sistema de dos entradas y una salida descrito por la siguiente matriz de transferencia:*

$$G(s) = \begin{bmatrix} \dfrac{1}{s+0{,}5} & \dfrac{1}{s} \end{bmatrix}$$

*y sujeto a unas restricciones sobre las acciones de control* $\bar{\mathbb{U}} = \mathbb{U}_1 \cup \mathbb{U}_2$ *donde:*

$$\mathbb{U}_1 := \left\{ u \in \mathbb{R}^2 \left| \begin{bmatrix} 0 & -1 \\ -1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix} u \leq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 2 \end{bmatrix} \right. \right\}$$

$$\mathbb{U}_2 := \left\{ u \in \mathbb{R}^2 \left| \begin{bmatrix} -1 & -1 \\ -1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} u \leq \begin{bmatrix} -2 \\ -1 \\ 3 \\ 3 \end{bmatrix} \right. \right\}$$

*El espacio de restricciones así definido es claramente no convexo, como se muestra en la figura 4.6.*

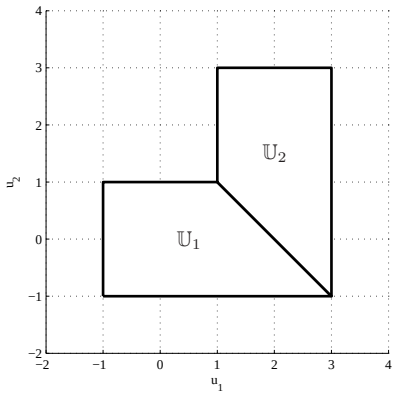


***Figura 4.6:*** Ejemplo 4.6. Restricciones sobre las acciones de control.

*Se supone que el sistema está muestreado a un período* $T_s = 1s$*, obteniéndose la representación en espacio de estados:*

$$A = \begin{bmatrix} 0{,}6065 & 0 \\ 0 & 1 \end{bmatrix} \quad B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad C = \begin{bmatrix} 0{,}7869 & 1 \end{bmatrix}$$

*Los parámetros de diseño del controlador se escogen de la siguiente manera:*

$$M = 1, \quad N = 10, \quad Q = 1, \quad R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

*La matriz de ponderación del estado final, $P$, se obtiene resolviendo la ecuación algebraica de Ricatti* (2.42)*:*

$$P = \begin{bmatrix} 0{,}7127 & 0{,}9620 \\ 0{,}9620 & 1{,}486 \end{bmatrix}$$

*Inicialmente, es necesario escribir el problema de optimización a resolver de la forma* (3.5)*. Para ello, en primer lugar se obtienen las matrices $H$ y $F$ mediante las expresiones* (2.5) *y* (2.6) *respectivamente:*

$$H = \begin{bmatrix} 1{,}980 & 1{,}988 \\ 1{,}988 & 11{,}49 \end{bmatrix} \quad F = \begin{bmatrix} 0{,}5942 & 1{,}988 \\ 1{,}206 & 10{,}49 \end{bmatrix}$$

*En cuanto a las regiones de restricciones $T_i$, dado que el horizonte de control elegido es $M = 1$ y que sólo existen restricciones sobre las acciones de control, se tiene $T_i = \mathbb{U}_i$ (y, por tanto, $\gamma = 2$).*
*De esta forma, se tienen dos subproblemas de optimización con restricciones lineales, sobre los que es posible aplicar cualquiera de las técnicas de* mpQP *y obtener la solución explícita. Las figuras 4.7 y 4.8 representan las particiones del estado, cada una de ellas de 9 regiones, correspondientes a ambos controladores explícitos.*

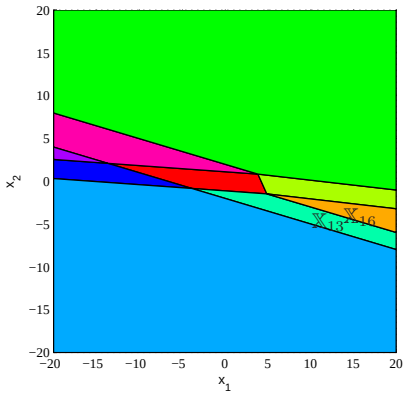


***Figura 4.7:*** MPC explicito, $u \in \mathbb{U}_1$

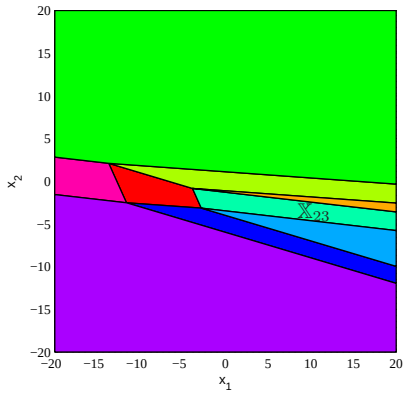


***Figura 4.8:*** MPC explicito, $u \in \mathbb{U}_2$

*Intersectando estas dos particiones mediante el procedimiento descrito en el algoritmo 4.2, se obtiene una nueva partición de 22 regiones no vacías y sin redundancias (de las 81 regiones posibles)(figura 4.9).*

*Como se ha visto, cada una de estas 22 regiones tiene dos posibles soluciones afines diferentes, cada una de ellas válida a un lado de la curva cuadrática definida por* (4.5)*. En este punto, es necesario decidir cuáles de estas regiones es necesario dividir resolviendo, para cada una de ellas, el problema* (4.6)*. Nótese que, con objeto de obtener una visualización sencilla, el ejemplo propuesto es muy simple y todos los problemas a resolver son casos particulares, puesto*

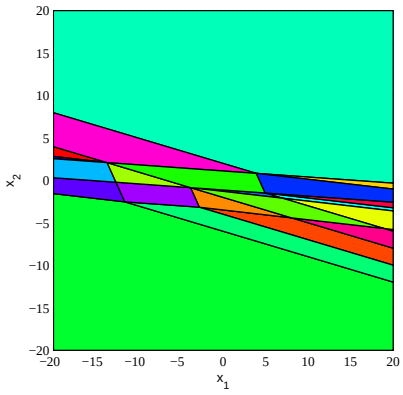


***Figura 4.9:*** Intersección de particiones.

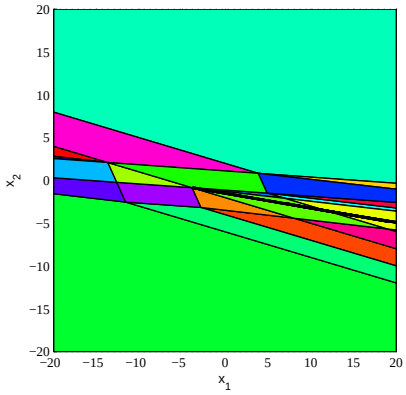


***Figura 4.10:*** Curvas divisoras.

*que las caras activas son siempre de dimensión* $[1 \times 2]$ *o* $[2 \times 2]$*. Si hubiera sido necesario seguir el procedimiento general descrito en el algoritmo 4.3, se habrían tenido que resolver 22 problemas de programación SOS.*
*De los 22 problemas resueltos, sólo dos de ellos son factibles, los correspondientes a* $\mathbb{X}_{13} \cap \mathbb{X}_{23}$ *y* $\mathbb{X}_{16} \cap \mathbb{X}_{23}$ *y por tanto es necesario dividir cada una de dichas intersecciones en dos subregiones. La figura 4.10 muestra estas curvas divisoras.*
*Por último, a la vista de que 20 de las regiones lineales no se han dividido, es de esperar que muchas de ellas puedan volver a unirse, por tener la misma solución afín. Si se aplica el algoritmo 4.4, se obtiene una partición final del estado de 17 regiones en total: 15 regiones lineales, con dos de ellas subdivididas por sus respectivas curvas (figura 4.11). Las ecuaciones* (4.13a) *y* (4.13b) *muestran la expresión explícita del controlador obtenido.*

$$\mathcal{K}(x) = \begin{cases} \begin{bmatrix} -0{,}24 & -0{,}11 \\ -0{,}064 & -0{,}89 \end{bmatrix} x + \begin{bmatrix} 0{,}0 \\ 0{,}0 \end{bmatrix} & si \quad \begin{bmatrix} 0{,}29 & 0{,}96 \\ -0{,}29 & -0{,}96 \\ 0{,}11 & 0{,}99 \\ -0{,}11 & -0{,}99 \end{bmatrix} x \leq \begin{bmatrix} -0{,}0043 \\ 1{,}9 \\ -1{,}3 \\ 3{,}5 \end{bmatrix} \\ & x^T \begin{bmatrix} -0{,}052 & 0{,}50 \\ 0{,}50 & 7{,}6 \end{bmatrix} x + [0{,}45\ 17] x + 7{,}9 \leq 0 \\ \begin{bmatrix} -0{,}24 & -0{,}11 \\ -0{,}064 & -0{,}89 \end{bmatrix} x + \begin{bmatrix} 0{,}0 \\ 0{,}0 \end{bmatrix} & si \quad \begin{bmatrix} 0{,}29 & 0{,}96 \\ -0{,}29 & -0{,}96 \\ 0{,}11 & 0{,}99 \\ -0{,}11 & -0{,}99 \end{bmatrix} x \leq \begin{bmatrix} -0{,}0043 \\ 1{,}9 \\ -1{,}3 \\ 3{,}5 \end{bmatrix} \\ & x^T \begin{bmatrix} -0{,}052 & 0{,}50 \\ 0{,}50 & 7{,}6 \end{bmatrix} x + [0{,}45\ 17] x + 7{,}9 \geq 0 \\ \begin{bmatrix} 0{,}0 & 0{,}0 \\ -0{,}10 & -0{,}91 \end{bmatrix} x + \begin{bmatrix} -1{,}0 \\ 0{,}17 \end{bmatrix} & si \quad \begin{bmatrix} -0{,}29 & -0{,}96 \\ 0{,}11 & 0{,}99 \\ -0{,}11 & -0{,}99 \\ 1{,}0 & 0{,}0 \end{bmatrix} x \leq \begin{bmatrix} 0{,}0043 \\ -1{,}3 \\ 3{,}5 \\ 20{,}0 \end{bmatrix} \\ & x^T \begin{bmatrix} 0{,}13 & 1{,}1 \\ 1{,}1 & 9{,}6 \end{bmatrix} x + [0{,}45\ 17] x + 7{,}9 \leq 0 \\ \vdots & \vdots \quad \vdots \end{cases} \tag{4.13a}$$

$$
\mathcal{K}(x) = \left\{ \begin{array}{lll}
\vdots & \vdots & \vdots \\
\begin{bmatrix} 0{,}064 & 0{,}90 \\ -0{,}064 & -0{,}90 \end{bmatrix} x + \begin{bmatrix} 2{,}0 \\ -0{,}0019 \end{bmatrix} & si & \begin{bmatrix} -0{,}29 & -0{,}96 \\ 0{,}11 & 0{,}99 \\ -0{,}11 & -0{,}99 \\ 1{,}0 & 0{,}0 \end{bmatrix} x \leq \begin{bmatrix} 0{,}0043 \\ -1{,}3 \\ 3{,}5 \\ 20{,}0 \end{bmatrix} \\
 & & x^T \begin{bmatrix} 0{,}13 & 1{,}1 \\ 1{,}1 & 9{,}6 \end{bmatrix} x + [0{,}45\ 17] x + 7{,}9 \geq 0 \\
\begin{bmatrix} -0{,}24 & -0{,}11 \\ -0{,}064 & -0{,}89 \end{bmatrix} x + \begin{bmatrix} 0{,}0 \\ 0{,}0 \end{bmatrix} & si & \begin{bmatrix} 0{,}29 & 0{,}96 \\ -0{,}29 & -0{,}96 \\ 0{,}11 & 0{,}99 \\ -0{,}11 & -0{,}99 \end{bmatrix} x \leq \begin{bmatrix} -0{,}0043 \\ 1{,}9 \\ -1{,}3 \\ 3{,}5 \end{bmatrix} \\
\begin{bmatrix} 0{,}0 & 0{,}0 \\ -0{,}10 & -0{,}91 \end{bmatrix} x + \begin{bmatrix} -1{,}0 \\ 0{,}17 \end{bmatrix} & si & \begin{bmatrix} -0{,}29 & -0{,}96 \\ 0{,}11 & 0{,}99 \\ -0{,}11 & -0{,}99 \\ 1{,}0 & 0{,}0 \end{bmatrix} x \leq \begin{bmatrix} 0{,}0043 \\ -1{,}3 \\ 3{,}5 \\ 20{,}0 \end{bmatrix} \\
\begin{bmatrix} -0{,}30 & -1{,}0 \\ 0{,}0 & 0{,}0 \end{bmatrix} x + \begin{bmatrix} -1{,}0 \\ 1{,}0 \end{bmatrix} & si & \begin{bmatrix} 0{,}91 & 0{,}41 \\ -0{,}072 & -1{,}0 \\ 0{,}072 & 1{,}0 \\ -0{,}29 & -0{,}96 \end{bmatrix} x \leq \begin{bmatrix} 3{,}9 \\ 1{,}1 \\ 1{,}1 \\ 1{,}9 \end{bmatrix} \\
\begin{bmatrix} -0{,}30 & -1{,}0 \\ 0{,}0 & 0{,}0 \end{bmatrix} x + \begin{bmatrix} 1{,}0 \\ -1{,}0 \end{bmatrix} & si & \begin{bmatrix} 0{,}11 & 0{,}99 \\ -0{,}91 & -0{,}41 \\ 0{,}072 & 1{,}0 \\ -0{,}11 & -0{,}99 \\ 1{,}0 & 0{,}0 \end{bmatrix} x \leq \begin{bmatrix} 1{,}3 \\ -3{,}9 \\ 1{,}1 \\ 0{,}90 \\ 20{,}0 \end{bmatrix} \\
\begin{bmatrix} 0{,}0 & 0{,}0 \\ 0{,}0 & 0{,}0 \end{bmatrix} x + \begin{bmatrix} -1{,}0 \\ 1{,}0 \end{bmatrix} & si & \begin{bmatrix} 0{,}29 & 0{,}96 \\ 0{,}072 & 1{,}0 \\ -0{,}11 & -0{,}99 \end{bmatrix} x \leq \begin{bmatrix} -0{,}0043 \\ -1{,}1 \\ 1{,}3 \end{bmatrix} \\
\begin{bmatrix} 0{,}0 & 0{,}0 \\ 0{,}0 & 0{,}0 \end{bmatrix} x + \begin{bmatrix} -1{,}0 \\ -1{,}0 \end{bmatrix} & si & \begin{bmatrix} 0{,}29 & 0{,}96 \\ -0{,}29 & -0{,}96 \\ -0{,}072 & -1{,}0 \\ -1{,}0 & 0{,}0 \end{bmatrix} x \leq \begin{bmatrix} 1{,}9 \\ 1{,}9 \\ -1{,}1 \\ 20{,}0 \end{bmatrix} \\
\begin{bmatrix} -0{,}24 & -0{,}11 \\ -0{,}064 & -0{,}89 \end{bmatrix} x + \begin{bmatrix} 0{,}0 \\ 0{,}0 \end{bmatrix} & si & \begin{bmatrix} 0{,}29 & 0{,}96 \\ -0{,}29 & -0{,}96 \\ 0{,}11 & 0{,}99 \\ -0{,}11 & -0{,}99 \end{bmatrix} x \leq \begin{bmatrix} -0{,}0043 \\ 1{,}9 \\ -1{,}3 \\ 3{,}5 \end{bmatrix} \\
\begin{bmatrix} 0{,}0 & 0{,}0 \\ -0{,}10 & -0{,}91 \end{bmatrix} x + \begin{bmatrix} 1{,}0 \\ -0{,}17 \end{bmatrix} & si & \begin{bmatrix} -0{,}29 & -0{,}96 \\ 0{,}11 & 0{,}99 \\ -0{,}11 & -0{,}99 \\ 1{,}0 & 0{,}0 \end{bmatrix} x \leq \begin{bmatrix} 0{,}0043 \\ -1{,}3 \\ 3{,}5 \\ 20{,}0 \end{bmatrix} \\
\begin{bmatrix} -0{,}30 & -1{,}0 \\ 0{,}0 & 0{,}0 \end{bmatrix} x + \begin{bmatrix} -3{,}0 \\ 3{,}0 \end{bmatrix} & si & \begin{bmatrix} 0{,}91 & 0{,}41 \\ -0{,}072 & -1{,}0 \\ 0{,}072 & 1{,}0 \\ -0{,}29 & -0{,}96 \end{bmatrix} x \leq \begin{bmatrix} 3{,}9 \\ 1{,}1 \\ 1{,}1 \\ 1{,}9 \end{bmatrix} \\
\begin{bmatrix} 0{,}0 & 0{,}0 \\ -0{,}10 & -0{,}91 \end{bmatrix} x + \begin{bmatrix} 3{,}0 \\ -0{,}52 \end{bmatrix} & si & \begin{bmatrix} 0{,}11 & 0{,}99 \\ -0{,}91 & -0{,}41 \\ 0{,}072 & 1{,}0 \\ -0{,}11 & -0{,}99 \\ 1{,}0 & 0{,}0 \end{bmatrix} x \leq \begin{bmatrix} 1{,}3 \\ -3{,}9 \\ 1{,}1 \\ 0{,}90 \\ 20{,}0 \end{bmatrix} \\
\begin{bmatrix} 0{,}0 & 0{,}0 \\ 0{,}0 & 0{,}0 \end{bmatrix} x + \begin{bmatrix} 3{,}0 \\ -1{,}0 \end{bmatrix} & si & \begin{bmatrix} 0{,}29 & 0{,}96 \\ 0{,}072 & 1{,}0 \\ -0{,}11 & -0{,}99 \end{bmatrix} x \leq \begin{bmatrix} -0{,}0043 \\ -1{,}1 \\ 1{,}3 \end{bmatrix} \\
\vdots & \vdots & \vdots
\end{array} \right. \tag{4.13b}
$$

$$\mathcal{K}(x) = \begin{cases} \vdots & \vdots \\ \begin{bmatrix} 0{,}0 & 0{,}0 \\ 0{,}0 & 0{,}0 \end{bmatrix} x + \begin{bmatrix} 1{,}0 \\ 3{,}0 \end{bmatrix} & si \quad \begin{bmatrix} 0{,}29 & 0{,}96 \\ -0{,}29 & -0{,}96 \\ -0{,}072 & -1{,}0 \\ -1{,}0 & 0{,}0 \end{bmatrix} x \leq \begin{bmatrix} 1{,}9 \\ 1{,}9 \\ -1{,}1 \\ 20{,}0 \end{bmatrix} \\ \begin{bmatrix} 0{,}0 & 0{,}0 \\ 0{,}0 & 0{,}0 \end{bmatrix} x + \begin{bmatrix} 3{,}0 \\ 3{,}0 \end{bmatrix} & si \quad \begin{bmatrix} -0{,}29 & -0{,}96 \\ 0{,}11 & 0{,}99 \\ 1{,}0 & 0{,}0 \\ 0{,}072 & 1{,}0 \\ -0{,}11 & -0{,}99 \end{bmatrix} x \leq \begin{bmatrix} 0{,}0043 \\ -0{,}90 \\ 20{,}0 \\ -1{,}1 \\ 1{,}3 \end{bmatrix} \end{cases} \tag{4.13c}$$

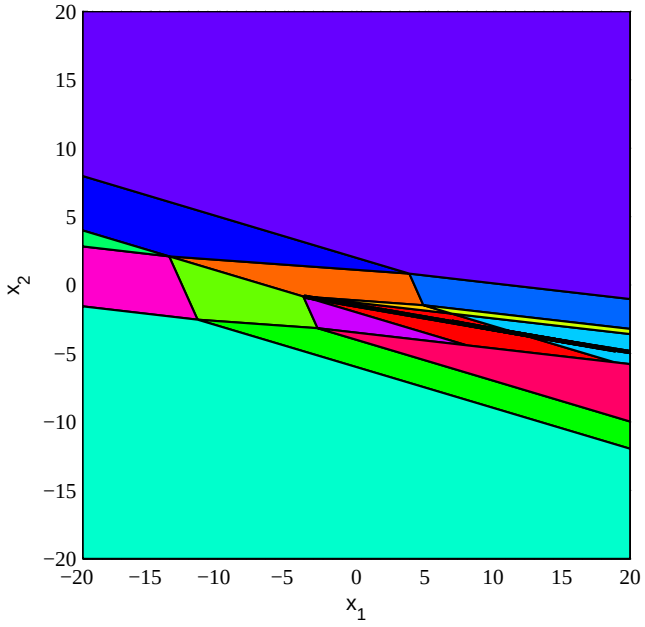


***Figura 4.11:*** Solución final.Partición del estado.

■

## 4.3. Metodología de intersección, división y unión para $\gamma > 2$

Cuando se tienen más de dos regiones convexas cuya unión define el espacio de restricciones no convexo, son necesarias nuevas consideraciones respecto a la metodología propuesta puesto que a priori existen diferentes opciones, dependiendo de en qué orden se realicen los diferentes pasos del algoritmo. Es decir, cuando se tienen más de dos particiones del estado, aunque la solución explícita

final sea única puede ser calculada de diferentes formas, siendo los algoritmos 4.5 y 4.6 las dos opciones principales.

**Algoritmo 4.5** Obtención de la solución explícita $\gamma > 2$. Algoritmo simultáneo

1. Obtener la solución explícita de los $\gamma$ problemas de optimización con restricciones convexas.
2. Intersectar todas las regiones de todas las particiones.
3. Para cada intersección no vacía, obtener las curvas correspondientes que dividen en varias subregiones.
4. Comprobar si en la región lineal hay más de una solución óptima y por tanto es necesario dividirlas.
5. Comprobar si es posible unir regiones no divididas (con una única solución).

En el algoritmo 4.5 la idea es obtener todas las regiones lineales factibles, comprobando la intersección de todas las combinaciones de regiones de las $\gamma$ particiones y, para cada una de ellas, obtener las curvas que indican cuando cada solución es mejor que las restantes. A priori, para cada una de las regiones lineales hay $\gamma$ posibles soluciones, pero puede que alguna de ellas se pueda descartar porque siempre sea mejor alguna de las otras soluciones.

En el algoritmo 4.6 por contra, se parte de dos particiones y se aplica el algoritmo 4.1, válido para $\gamma = 2$. Como resultado de éste se obtiene una nueva partición lineal del estado ($\mathbb{T}_2$) para la que algunas regiones (pertenecientes a $\mathbb{T}_2^1$) tienen una única solución afín, mientras que para otras (las de $\mathbb{T}_2^2$) existen dos posibles soluciones, dependiendo de a qué lado de la correspondiente curva se encuentre el vector de estados $x$. Sobre esta partición $\mathbb{T}_2$, se añade una nueva partición lineal, intersectando sus regiones. En este caso se obtendrán regiones con 1, 2 o hasta 3 posibles soluciones divididas por las correspondientes curvas. Añadiendo de manera sucesiva las $\gamma$ particiones, se llega a la solución explícita final.
Siguiendo esta nomenclatura, se definen los conjuntos $\mathbb{T}_i$ como las particiones resultantes tras intersectar las particiones de $i$ subproblemas, y los subconjuntos $\mathbb{T}_i^j$ como las regiones de $\mathbb{T}_i$ que tienen $j$ soluciones diferentes.

Entre los dos algoritmos propuestos podrían diseñarse muchos más con una filosofía intermedia basada en formar subgrupos de particiones, obtener la solución explícita para cada uno de ellos e intersectarlos posteriormente.

Si bien todos los métodos van a dar con la misma solución explícita final, puesto que ésta es única, utilizar uno u otro va a suponer un coste del algoritmo fuera de línea que puede ser significativamente diferente, por lo que, a continuación, se analiza dicho coste para cada uno de los algoritmos propuestos.

**Algoritmo 4.6** Obtención de la solución explícita $\gamma > 2$. Algoritmo progresivo

1. Obtener la solución explícita de los $\gamma$ problemas de optimización.
2. Tomar la partición con menor número de regiones e inicializar:
   - $\mathbb{T}_1^1 = \{X_{1i}\}$, $\mathbb{T}_1^2 = \ldots = \mathbb{T}_1^\gamma = \{\emptyset\}$.
   - $\mathbb{T}_1 = \mathbb{T}_1^1$.
   - $i = 2$.
3. Tomar la siguiente partición con menor número de regiones, $\mathbb{X}_i$.
4. Intersectar con las regiones de cada una de las particiones anteriores: $\mathbb{W}_i^j = \{X_{jk} \cap X_{il} | X_{jk} \in \mathbb{T}_{i-1}^j, X_{il} \in \mathbb{X}_i\}$.
5. Para cada región de $\mathbb{W}_i^j$,comprobar cuáles de las $j+1$ posibles soluciones son aplicables a la región.
6. Actualizar las listas de regiones, $\mathbb{T}_i^j$ en función del número de soluciones aplicables.
7. Si es posible, unir las regiones de $\mathbb{T}_i^j$ con el mismo conjunto de soluciones.
8. $i = i + 1$.
9. Si $i \leq \gamma$, volver a 3.

### 4.3.1. Análisis del coste de los algoritmos fuera de línea

Dado que los algoritmos descritos son bastante complejos, resulta complicado analizar su coste en términos de operaciones aritméticas elementales. En lugar de ello, una opción es cuantificar el número de problemas de optimización de diferentes tipos que es necesario resolver. Aunque el tiempo que se tarda en resolver cada problema de un mismo tipo no es constante (puesto que depende de muchos factores como el número de variables, número de restricciones, etc...) sí que se mantiene dentro de un mismo orden de magnitud, y por lo tanto puede ser suficiente para comparar los diferentes algoritmos.
En el anexo A se realiza una comparación en estos términos entre el algoritmo simultáneo 4.5 y el progresivo 4.6.

En dicho anexo, se muestra como a priori ninguno de los dos algoritmos es menos costoso para todos los casos posibles, sino que depende en gran medida de cuantas de las regiones de intersección tienen más de una solución factible, es decir, de los poliedros de restricciones. No obstante, sí puede observarse que, para diferentes hipótesis planteadas, si el valor de $\gamma$ no es muy elevado, usualmente el algoritmo menos costoso será el progresivo. Ante la resolución de un problema real, no es realista plantear un número de conjuntos de restricciones $\gamma$ muy alto, por lo que se aplicará el algoritmo progresivo si no se dispone de

ninguna información adicional.

### 4.3.2. Descripción del procedimiento

En el apartado anterior se ha mostrado como, en general, la forma más conveniente de obtener la solución explícita del controlador es la mostrada en el algoritmo 4.6. Esencialmente, dicho algoritmo obtiene las $\gamma$ particiones del espacio de estados y las va intersectando una a una. Cada vez que se añade una partición, por tanto, se realizan una serie de pasos que son muy similares a los que se llevan a cabo en el algoritmo presentado para dos regiones no convexas: intersectar las regiones de las dos particiones, dividirlas con curvas cuadráticas si es necesario y unir, cuando sea posible, las regiones que tienen una misma solución. No obstante, algunos de estos pasos presentan algunas particularidades que se describen a continuación.

#### Obtención de la solución explícita de los subproblemas

Al igual que en el caso de $\gamma = 2$, la solución explícita a cada uno de los subproblemas de control predictivo con restricciones lineales puede obtenerse con cualquiera de los métodos expuestos en la sección 2.3.

#### Intersección de las particiones de estado

En cada paso del algoritmo tendremos dos particiones lineales del espacio de estados: $\mathbb{T}_{i-1}$, cuyas regiones pueden estar divididas por curvas cuadráticas, proveniente de pasos anteriores del algoritmo, y $\mathbb{X}_i$, obtenida como solución a un problema con restricciones lineales y añadida en el paso actual.
Aunque para el desarrollo del algoritmo 4.6 se haya dividido la partición $\mathbb{T}_{i-1}$ en diferentes subconjuntos formados por las regiones con diferente número de posibles soluciones afines, a efectos prácticos en este paso hay que intersectar todas las regiones de dos particiones lineales del espacio de estados. Por tanto, la factibilidad de estas intersecciones puede determinarse resolviendo un problema LP, de igual forma que para el caso de $\gamma = 2$:

$$\begin{aligned} &\min\ s \\ &\text{sujeto a:}\quad \begin{bmatrix} L_{Ti} \\ L_{Xj} \end{bmatrix} x - \begin{bmatrix} W_{Ti} \\ W_{Xj} \end{bmatrix} \leq \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} s \end{aligned}$$

En cuanto a la eliminación de redundancias en la definición de las regiones, las proposiciones 4.2 y 4.3, aplicables a regiones adyacentes y contiguas, son igualmente válidas. No obstante, el procedimiento en este caso sí difiere en el modo de obtener qué regiones son adyacentes y contiguas. Para la partición $\mathbb{X}_i$, sigue siendo posible aplicar el teorema 4.1 o el algoritmo 2.2, que permiten respectivamente decidir cuando una región tiene una única región contigua a

través de una de sus facetas o en caso de no ser así, obtener las adyacentes. Sin embargo, para la partición $\mathbb{T}_{i-1}$ esto no es posible, puesto que las regiones ya no se corresponden con regiones críticas asociadas a diferentes conjuntos de restricciones activas de un mismo problema. Por tanto, es necesario diseñar otro método para detectar qué regiones son adyacentes y contiguas. Para ello, se formulan las siguientes proposiciones.

**Proposición 4.5.** *Dadas dos regiones diferentes de una misma partición lineal* $\mathbb{X}_i$, $X_1 := \{x \in \mathbb{R}^n | L_1 x \leq W_1\}$ *y* $X_2 := \{x \in \mathbb{R}^n | L_2 x \leq W_2\}$*:*

- $dim(X_1 \cap X_2) = n - 1$ *si* $X_1$ *y* $X_2$ *son adyacentes.*
- $dim(X_1 \cap X_2) < n - 1$ *si* $X_1$ *y* $X_2$ *no son adyacentes.*

□

**Prueba.** Al pertenecer ambas regiones a una misma partición, como se vio en (2.28) $\text{int}(X_i) \cap \text{int}(X_j) = \emptyset$ y por tanto $\dim(X_1 \cap X_2) < n$. Teniendo en cuenta que, por definición, dos regiones son adyacentes si $\dim(X_1 \cap X_2) = n - 1$, el lema queda demostrado. ■

**Proposición 4.6.** *Dos regiones de una partición* $\mathbb{T}_i = \mathbb{T}_{i-1} \cap \mathbb{X}_i$*:*

$$T_1^i = T_1^{i-1} \cap X_1$$
$$T_2^i = T_2^{i-1} \cap X_2$$

*serán adyacentes si y sólo si se cumple una de estas tres condiciones:*

- $T_1^{i-1} = T_2^{i-1}$ *y* $X_1$ *es adyacente a* $X_2$ *con una intersección* $S_1 = X_1 \cap X_2$ *que cumple* $dim(S_1 \cap T_1^{i-1}) = n - 1$*.*
- $X_1 = X_2$ *y* $T_1^{i-1}$ *es adyacente a* $T_2^{i-1}$ *con una intersección* $S_2 = T_1^{i-1} \cap T_2^{i-1}$ *que cumple* $dim(S_2 \cap X_1) = n - 1$*.*
- $T_1^{i-1}$ *es adyacente a* $T_2^{i-1}$ *a través de un hiperplano* $h$*,* $X_1$ *es adyacente a* $X_2$ *por el mismo hiperplano y* $dim(S_1 \cap S_2) = n - 1$*.*

□

**Prueba.** Si $T_1^i$ y $T_2^i$ son adyacentes, por definición su intersección tiene dimensión $n - 1$. Podemos reescribir dicha intersección como:

$$T_1^i \cap T_2^i = (T_1^{i-1} \cap X_1) \cap (T_2^{i-1} \cap X_2) = (X_1 \cap X_2) \cap (T_1^{i-1} \cap T_2^{i-1}) = S_1 \cap S_2$$

Demostraremos en primer lugar la necesidad de las condiciones expuestas. Para que $T_1^i \cap T_2^i$ sea de dimensión $n - 1$, $S_1$ y $S_2$ tienen que ser, como mínimo, de dimensión $n - 1$. Se pueden distinguir tres casos diferentes:

- $\dim(S_1) = n - 1$ y $\dim(S_2) = n$. Por la proposición 4.5, $T_1^{i-1}$ y $T_2^{i-1}$ tienen que ser la misma región y $X_1$ y $X_2$ adyacentes. Para que $T_1^i \cap T_2^i$ no esté vacía, se debe cumplir también $S_1 \cap X_2 \neq \emptyset$.

- $\dim(S_1) = n$ y $\dim(S_2) = n-1$. Por la proposición 4.5, $T_1^{i-1}$ y $T_2^{i-1}$ tienen que ser adyacentes y $X_1 = X_2 = S_1$ la misma región. En este caso, para que $T_1^i \cap T_2^i$ no esté vacía, se debe cumplir $S_2 \cap X_1 \neq \emptyset$.
- $\dim(S_1) = \dim(S_2) = n-1$. Como se ha visto en la proposición 4.5, esto implica que $T_1^{i-1}$ y $T_1^{i-1}$ son adyacentes y que $X_1$ y $X_2$ también lo son. Además, la intersección de las dos intersecciones $S_1 \cap S_2$ sólo será de dimensión $n-1$ si corresponde al mismo hiperplano.
- $\dim(S_1) = n$ y $\dim(S_2) = n$. En este caso, por la proposición 4.5 $T_1^{i-1} = T_2^{i-1}$ y $X_1 = X_2$, por lo que $T_1^i = T_2^i$ y se trata de un caso trivial.

En cuanto a la suficiencia, hay que probar que si se cumplen las condiciones de cada uno de los tres supuestos, $T_1^i$ y $T_2^i$ serán adyacentes, o lo que es lo mismo que dim $(T_1^i \cap T_2^i) = n-1$.

- Si $T_1^{i-1} = T_2^{i-1}$, $X_1$ y $X_2$ son adyacentes y $\dim(S_1 \cap T_1^{i-1}) = n-1$:

  $$T_1^i \cap T_2^i = S_1 \cap S_2 = S_1 \cap T_1^{i-1}$$

  Por tanto, $\dim(T_1^i \cap T_2^i) = n-1$
- Si $X_1 = X_2$, $T_1^{i-1}$ y $T_2^{i-1}$ son adyacentes y $\dim(S_2 \cap X_1) = n-1$:

  $$T_1^i \cap T_2^i = S_1 \cap S_2 = S_2 \cap X_1$$

  Por tanto, $\dim(T_1^i \cap T_2^i) = n-1$
- Si $dim(S_1 \cap S_2) = n-1$, es trivial que las regiones son adyacentes.

■

**Ejemplo 4.7.** *Las figuras 4.12 y 4.13 muestran respectivamente las regiones de dos particiones lineales, $\mathbb{X}$ y $\mathbb{T}^1$. Si se intersectan ambas particiones, se obtiene la partición $\mathbb{T}^2$ (figura 4.14). En ésta se pueden observar regiones adyacentes de los tres tipos diferentes contemplados en la proposición 4.6:*

- $T_1^2 = X_1 \cap T_1^1$ *es adyacente a* $T_2^2 = X_1 \cap T_2^1$.
- $T_1^2$ *es adyacente a* $T_3^2 = X_2 \cap T_1^1$.
- $T_2^2$ *es adyacente a* $T_5^2 = X_3 \cap T_3^1$ *puesto que* $X_1$ *es adyacente a* $X_3$ *a través del mismo hiperplano que* $T_2^1$ *es adyacente a* $T_3^1$.

■

Es importante notar que, para los dos primeros supuestos de la proposición 4.6, la comprobación de si dos regiones $T_1^i$ y $T_2^i$ son adyacentes se puede realizar de manera sencilla en la iteración anterior, cuando se han eliminado las redundancias de dichas regiones. Supongamos que se tienen dos particiones,

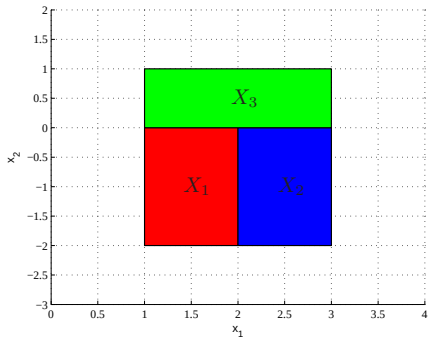


***Figura 4.12:*** Regiones de la partición $\mathbb{X}$.

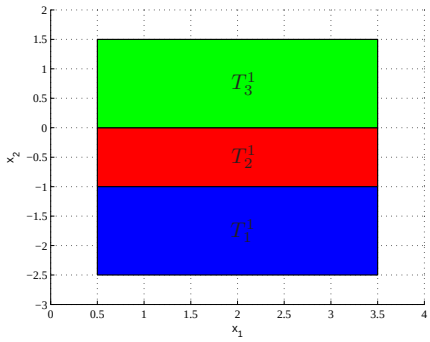


***Figura 4.13:*** Regiones de la partición $\mathbb{T}^1$.

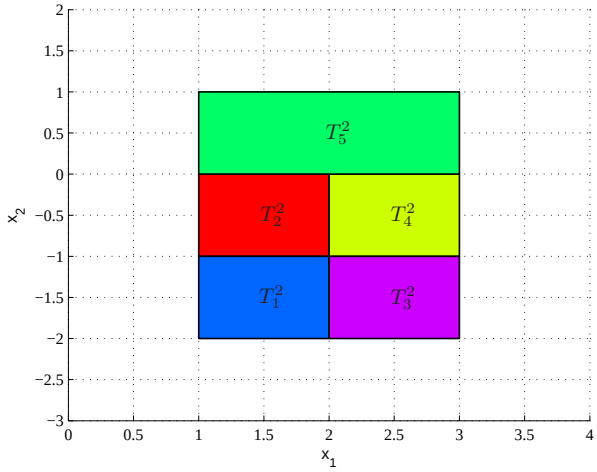


***Figura 4.14:*** Regiones de la partición $\mathbb{T}^2 = \mathbb{X} \cap \mathbb{T}^1$.

$\mathbb{X}_1$ y $\mathbb{X}_2$ y que se parte de las regiones $X_{11}$ y $X_{12}$ que son adyacentes, y de una región $X_{21}$ que, tras resolver el problema de factibilidad, se comprueba que tiene intersección no nula con las anteriores. Si se sigue el procedimiento que se ha descrito en el algoritmo 4.2, para eliminar las posibles redundancias de $X_{11} \cap X_{21}$ y de $X_{12} \cap X_{21}$ es necesario comprobar el hiperplano $h$ correspondiente a la faceta común a ambas regiones. Es evidente que el supuesto de la proposición, $\dim(X_{11} \cap X_{12} \cap X_{21}) = n - 1$ es cierto si y sólo si $h$ no es redundante en las dos regiones.

Por tanto, cuando se comprueba la redundancia de un hiperplano en dos regiones $(X_1 \cap T_1^{i-1})$ y $(X_2 \cap T_1^{i-1})$, o bien se elimina de al menos una de ellas (en cuyo caso no es posible que sean adyacentes a través del hiperplano en cuestión), o bien las dos regiones son adyacentes.

En cuanto al tercer supuesto de la proposición, representa un tipo de regio-

nes adyacentes que sólo se producirá en casos muy particulares, cuando haya hiperplanos coincidentes en la descripción de las diferentes zonas de restricciones. Si esto se detecta, puede formularse un LP con las desigualdades de todas las regiones que intervienen para obtener $S_1 \cap S_2$ y comprobar su dimensión.

**Proposición 4.7.** *Dos regiones de una partición $\mathbb{T}_i = \mathbb{T}_{i-1} \cap \mathbb{X}_i$:*

$$T_1^i = T_1^{i-1} \cap X_1$$
$$T_2^i = T_2^{i-1} \cap X_2$$

*serán contiguas si se cumple una de estas tres condiciones:*

- *$T_1^{i-1} = T_2^{i-1}$ y $X_1$ es contigua a $X_2$ con una faceta común $\mathcal{F}_1$ que cumple $dim(\mathcal{F}_1 \cap T_1^{i-1}) = n - 1$.*
- *$X_1 = X_2$ y $T_1^{i-1}$ es contigua a $T_2^{i-1}$ con una faceta común $\mathcal{F}_2$ que cumple $dim(\mathcal{F}_2 \cap X_1) = n - 1$.*
- *$X_1$ es contigua a $X_2$ con una faceta común $\mathcal{F}_1$, $T_1^{i-1}$ es contigua a $T_2^{i-1}$ con una faceta común $\mathcal{F}_2$ y $dim(\mathcal{F}_1 \cap \mathcal{F}_2) = n - 1$.*

□

**Prueba.** Resulta trivial a partir de la demostración de 4.6 y la definición de región contigua. ■

Nótese que, a diferencia de la proposición 4.6, la 4.7 introduce sólo una condición suficiente, pero no necesaria. Por tanto, puede haber más regiones contiguas de las que se encontrarán con la proposición. No obstante, esto no es un problema demasiado grave puesto que, dado que regiones contiguas son también adyacentes, estas regiones contiguas serán identificadas y tratadas como tal, y también serán eliminadas sus redundancias (aunque puede que a un coste un poco superior que si se supiera que son contiguas).

Una vez obtenidas que regiones son adyacentes y contiguas, es posible aplicar el algoritmo 4.2 a partir del paso 3, lo que permite obtener todas las regiones lineales sin redundancias.

Por último, una vez se ha realizado la intersección de regiones, es conveniente aplicar de nuevo el algoritmo de unión 4.4 para las regiones con la misma solución, ya que si alguna se puede unir se reduce el número de problemas de suma de cuadrados a realizar.

#### Eliminación de las soluciones no óptimas

Tras obtener la intersección de las regiones lineales y eliminar sus redundancias, nos encontramos en el punto 5 del algoritmo 4.6, con una serie de conjuntos de regiones $\mathbb{W}_i^j = \{X_{jk} \cap X_{il} | X_{jk} \in \mathbb{T}_{i-1}^j, X_{il} \in \mathbb{X}_i\}$. Las regiones $X_{jk}$, son aquellas que en la partición anterior tenían $j$ posibles soluciones por lo

que, al intersectarlas con una región de la nueva partición, $X_{il}$, a priori tienen $j+1$ posibles soluciones.

Al igual que se vio en la sección 4.2.3, es importante, desde el punto de vista del coste computacional del algoritmo en línea, decidir cuando la región de intersección tiene efectivamente $j+1$ posibles soluciones o cuando son menos, porque algunas de ellas quedan descartadas al ser peor que las demás para todos los puntos de la región lineal.

Al igual que cuando se tienen únicamente dos soluciones, la opción más sencilla es obtener los mínimos y máximos de cada uno de los índices de coste, resolviendo un $QP$ para los primeros y evaluando la función en los vértices para los segundos.

Dada una determinada región lineal de $\mathbb{W}_i^j$:

$$X_r := \{x \in \mathbb{R}^n | L_r x \leq W_r\}$$

Se puede eliminar una de las soluciones afines $k$, si se cumple:

$$\left\{k | \exists l \in \{1 \dots (j+1)\} \backslash \{k\} : \min(V_k^{OPT}) \geq \max(V_l^{OPT})\right\} \tag{4.14}$$

puesto que al menos la solución $l$ va a proporcionar un coste menor que la $k$. Por tanto, la solución $k$ pasará a formar parte del conjunto de soluciones no óptimas, $\mathcal{NO}(X_r) = \mathcal{NO}(X_r) \cup k$.

Una vez eliminadas de esta manera todas las soluciones posibles, para decidir cuando una de las soluciones afines posibles, $k$, es una de las factibles en $X_r$ es necesario comprobar si, en algún subconjunto de dicha región, el índice de coste asociado a la solución afín, $V_k^{OPT}$, es menor que todos los índices de coste asociados al resto de soluciones. Para ello, se ha de resolver un problema de consistencia de la siguiente forma:

$$\begin{cases} L_r x \leq W_r \\ V_k^{OPT} \leq V_l^{OPT} \quad l = \{1 \dots (j+1)\} \backslash \{\mathcal{NO}(X_r) \cup k\} \end{cases} \tag{4.15}$$

Si el problema anterior es consistente, existe algún subconjunto de $X_r$ en los que el menor índice de coste es el asociado a la solución $k$, y por tanto dicha solución es una de las posibles.

El problema (4.15) es conceptualmente igual a (2.49), planteado para el caso de $\gamma = 2$, y por tanto puede resolverse como un problema de suma de cuadrados. En este caso no existen restricciones de igualdad ($f_i(x)$) las restricciones de desigualdad $g_i(x)$ son de dos tipos:

- Desigualdades lineales que definen $X_r$ (tantas como filas tienen las matrices $L_r$ y $W_r$).
- $j$ curvas cuadráticas de la forma $V_l^{OPT}(x) - V_k^{OPT}(x) \geq 0$.

Una vez resuelto un problema SOS de la forma (4.15), se sabe si una determinada solución afín, $k$, es una de las factibles en la región. Para comprobar todas las soluciones será por tanto necesario resolver $j+1$ problemas.

Es interesante tener en cuenta el caso particular de $j = 1$, puesto que las regiones de $\mathbb{W}_i^1$ se forman intersectando las regiones pertenecientes a $\mathbb{T}_{i-1}^1$ con las pertenecientes a $\mathbb{X}_i$. Por definición, las regiones pertenecientes a dichas particiones tienen una única solución afín en cada una de ellas. Si se plantea el problema de forma general, será necesario resolver dos problemas SOS:

$$\left\{ \begin{array}{c} L_r x \leq W_r \\ V_1^{OPT} \leq V_2^{OPT} \end{array} \right. \qquad \left\{ \begin{array}{c} L_r x \leq W_r \\ V_1^{OPT} \geq V_2^{OPT} \end{array} \right.$$

No obstante, como ya se vio en la sección 4.2.3, es menos costoso computacionalmente resolver un único SOS con restricción de igualdad (4.16) y, en el caso de que el problema no sea consistente, comprobar después cuál de las dos soluciones es la óptima en la región.

$$\left\{ \begin{array}{c} L_r x \leq W_r \\ V_1^{OPT} = V_2^{OPT} \end{array} \right. \tag{4.16}$$

Otra consideración a tener en cuenta que, si bien no reduce el número de SOS a resolver, sí hace que algunos de estos puedan ser más sencillos, es ir progresivamente eliminando de las comprobaciones aquellas soluciones que no son factibles. Es decir, si se resuelve un primer problema (4.17) y se obtiene que el problema no es factible, se tiene la certeza de que la primera solución nunca va a ser la mejor en la región $X_r$. Por tanto, en el siguiente SOS a resolver puede eliminarse una de las restricciones cuadráticas (4.18).

$$\left\{ \begin{array}{ll} L_r x \leq W_r & \\ V_1^{OPT} \leq V_l^{OPT} & l = \{2 \dots (j+1)\} \end{array} \right. \tag{4.17}$$

$$\left\{ \begin{array}{ll} L_r x \leq W_r & \\ V_2^{OPT} \leq V_l^{OPT} & l = \{3 \dots (j+1)\} \end{array} \right. \tag{4.18}$$

En el problema (4.18) se ha eliminado la restricción $V_2^{OPT} \leq V_1^{OPT}$, lo que en general hará que sea menos costoso, puesto que habrá que definir un polinomio SOS menos y por tanto el problema SDP a resolver tendrá menos variables.

El algoritmo 4.7 resume el procedimiento para obtener qué soluciones son factibles para $j < 1$ (para $j = 1$, como se ha visto, es aplicable el algoritmo 4.3).

Con este procedimiento se ha resuelto el punto 5 del algoritmo 4.6. El punto 6 es directo, puesto que consiste simplemente en reasignar las regiones en $\mathbb{W}_i^j$ en los diferentes $\mathbb{T}_i^j$ en función de cuantas de las soluciones han resultado ser factibles.

#### Minimización del número de regiones

Por último, el paso 7 del algoritmo 4.6 consiste en unir aquellas regiones contiguas con una misma solución afín (que por tanto debe ser única en cada una de ellas). Para ello, se aplica el algoritmo 4.8 que es directamente análogo al 4.4 descrito en la sección 4.2.4.

**Algoritmo 4.7** Eliminación de las soluciones no óptimas.

1. Para cada región $X_r = T_{i-1} \cap X_i$ obtener el conjunto de $j+1$ soluciones posibles como la unión de las soluciones factibles en $T_{i-1}$ y $X_i$.
2. Inicializar el conjunto de soluciones no óptimas: $\mathcal{NO}(X_r) = \emptyset$.
3. Obtener los máximos y mínimos de cada índice de coste, $\max(V_{iN}^{OPT})$, $\min(V_{iN}^{OPT})$.
4. Para $k = 1$ hasta $j+1$:
   *a*) Si $\exists l \in \{1 \dots (j+1)\} \backslash \{k\} : \min(V_k^{OPT}) \geq \max(V_l^{OPT})$, entonces $\mathcal{NO}(X_r) = \mathcal{NO}(X_r) \cup k$.
5. Obtener el conjunto de soluciones óptimas: $\mathcal{O}(X_r) = \{1 \dots (j+1)\} \backslash \mathcal{NO}(X_r)$
6. Para $k \in \mathcal{O}(X_r)$:
   *a*) Resolver el problema SOS:
   $$\begin{cases} L_r x \leq W_r \\ V_k^{OPT} \leq V_l^{OPT} \quad l = \{1 \dots (j+1)\} \backslash \{\mathcal{NO}(X_r) \cup k\} \end{cases}$$
   *b*) Si el problema no es factible, $\mathcal{NO}(X_r) = \mathcal{NO}(X_r) \cup k$.
7. Actualizar el conjunto de soluciones óptimas: $\mathcal{O}(X_r) = \mathcal{O}(X_r) \backslash \mathcal{NO}(X_r)$

**Algoritmo 4.8** Minimización del número de regiones lineales para $\gamma > 2$

1. Agrupar las regiones de $\mathbb{T}_i^1$ según su solución afín correspondiente.
2. Para cada grupo de regiones:
   - Obtener la disposición de planos $A$ como (4.11), eliminando desigualdades innecesarias.
   - Obtener marcas de todas las regiones según (4.12).
   - Aplicar algoritmo de unión de regiones:
     - Si no se desea solapamiento, aplicar algoritmo de ramificación y poda de [GTM08].
     - Si se permite solapamiento, aplicar algoritmo de minimización de lógica booleana de [GTM08].

**Ejemplo 4.8.** *Se desea obtener un controlador explícito para el sistema de dos entradas y una salida del ejemplo 4.6 muestreado con un período $T_s = 1s$. Dicho sistema puede ser descrito por su representación en espacio de estados:*

$$\begin{cases} x_{k+1} = Ax_k + Bu_k \\ y_k = Cx_k \end{cases}$$

$$A = \begin{bmatrix} 0{,}6065 & 0 \\ 0 & 1 \end{bmatrix} \quad B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad C = \begin{bmatrix} 0{,}7869 & 1 \end{bmatrix}$$

*El sistema además está sujeto a unas restricciones sobre los estados $\{-15 \leq x_1 \leq 15; -10 \leq x_2 \leq 10\}$ y sobre las acciones de control $\bar{\mathbb{U}} = \mathbb{U}_1 \cup \mathbb{U}_2$ donde:*

$$\mathbb{U}_1 := \left\{ u \in \mathbb{R}^2 \;\middle|\; \begin{array}{l} -1 \leq u_1 \leq 3 \\ -1 \leq u_2 \leq 1 \end{array} \right\} \quad \mathbb{U}_2 := \left\{ u \in \mathbb{R}^2 \;\middle|\; \begin{array}{l} 1 \leq u_1 \leq 3 \\ 1 \leq u_2 \leq 3 \end{array} \right\}$$

*El espacio de restricciones así definido es claramente no convexo, como se muestra en la figura 4.15.*

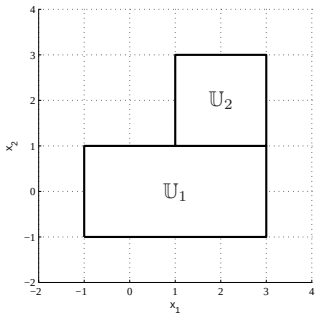


***Figura 4.15:*** Restricciones sobre acciones de control.

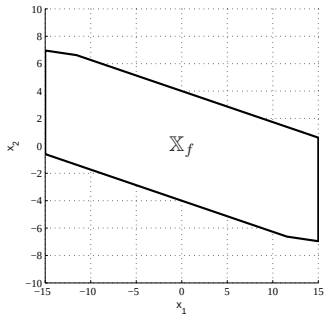


***Figura 4.16:*** Espacio de restricciones terminal del estado.

*Además, con objeto de garantizar la estabilidad del sistema en bucle cerrado, se añade una restricción terminal sobre el estado $x_N \in \mathbb{X}_f$ calculada siguiendo el procedimiento descrito en el capítulo 6 (figura 4.16).*

*Los parámetros de diseño del controlador se escogen de la siguiente manera:*

$$M = N = 5, \quad Q = 0{,}1, \quad R = I$$

*La matriz de ponderación del estado final, $P$, se obtiene resolviendo la correspondiente ecuación algebraica de Ricatti.*

*Inicialmente, es necesario escribir el problema de optimización a resolver de la forma (3.5). Para ello, se forman las regiones de restricciones $T_i$. Dado que el horizonte elegido es $M = N = 5$ y que sólo existen restricciones no convexas sobre las acciones de control, se tiene $\gamma = 2^5 = 32$).*
*De esta forma, se tienen 32 subproblemas de optimización con restricciones*

*lineales, sobre los que se resuelven los respectivos problemas* mpQP *obteniendo particiones del espacio de estados definidas con un número variable de regiones entre 13 y 42.*

*Aplicando el algoritmo 4.6, se obtiene la partición que se muestra en la figura 4.17. Dicha partición está formada por 210 regiones, 128 de ellas con una única solución, 70 con dos soluciones y 12 con tres.*

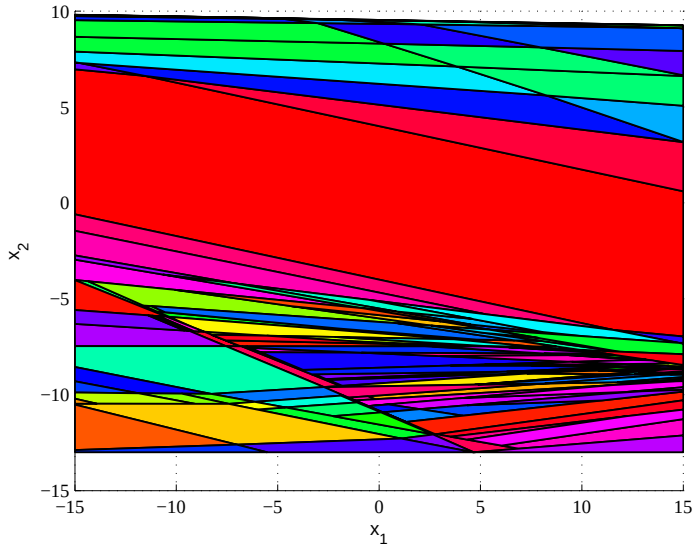


***Figura 4.17:*** Solución final $N = 5$. Partición del estado.

*Para la obtención de esta solución, ha sido necesaria la resolución de 6668 problemas de suma de cuadrados, y el procedimiento en su totalidad ha empleado 1686 segundos en un Intel core i5 750 a 2.66 GHz.* ■

## 4.4. Metodología de la envolvente convexa

En las secciones 4.2 y 4.3 se ha propuesto una metodología para la obtención de la solución explícita del problema de optimización (3.3):

$$\mathcal{P}_{N,M}(x): \quad V^{OPT}_{N,M}(x) := \min \tfrac{1}{2}\mathbf{u}^T H \mathbf{u} + \mathbf{u}^T F x,$$

$$\text{sujeto a:}$$
$$(\mathbf{u}, x) \in \mathbb{T}$$

donde:

$$\mathbb{T} = \bigcup_{i=0}^{\gamma} T_i$$

Para ello, se intersectan, dividen y unen las soluciones de los diferentes subproblemas (3.5):

$$\mathcal{P}_{N,M}(x): \quad V_{N,M}^{OPT}(x) := \min \ \left\{V_{iN}^{OPT}(x)\right\}$$

donde:

$$V_{iN}^{OPT}(x) := \min \ \tfrac{1}{2}\mathbf{u}^T H \mathbf{u} + \mathbf{u}^T F x,$$

sujeto a:
$(\mathbf{u}, x) \in T_i$

En la presente sección, se va a proponer un enfoque diferente para la obtención de la solución explícita, basado en el concepto de la envolvente convexa (definición 2.11).

Dados los $\gamma$ conjuntos de restricciones de nuestro problema (3.3), como en general éstas dependen del vector del espacio de estados, para poder calcular su envolvente convexa será necesario trabajar en el espacio de $(\mathbf{u}, x)$:

$$T_i := \left\{ (\mathbf{u}, x) \in \mathbb{R}^{Mm+n} \,\middle|\, \begin{bmatrix}\Phi_i & \Lambda_i\end{bmatrix} \begin{bmatrix}\mathbf{u} \\ x\end{bmatrix} \leq \Delta_i \right\} \tag{4.19}$$

Definimos la envolvente convexa de la unión de todos los conjuntos $T_i$, $\mathbb{T}$, como:

$$\mathbf{conv}(\mathbb{T}) = \left\{ (\mathbf{u}, x) \in \mathbb{R}^{Mm+n} \,\middle|\, \begin{bmatrix}\Phi_C & \Lambda_C\end{bmatrix} \begin{bmatrix}\mathbf{u} \\ x\end{bmatrix} \leq \Delta_C \right\}$$

Imponiendo estas nuevas restricciones convexas, puede definirse un nuevo problema de optimización:

$$\begin{aligned} \mathcal{P}_{N,M}(x): \quad & V_{N,M}^{OPT}(x) := \min \ \tfrac{1}{2}\mathbf{u}^T H \mathbf{u} + \mathbf{u}^T F x, \\ & \text{sujeto a:} \\ & (\mathbf{u}, x) \in \mathbf{conv}(\mathbb{T}) \end{aligned} \tag{4.20}$$

En adelante, por comodidad en la notación se nombran los problemas (3.3), (3.5) y (4.20) como $\mathcal{P}_{\mathbb{T}}$,$\mathcal{P}_{T_i}$ y $\mathcal{P}_{\mathbf{conv}(\mathbb{T})}$ respectivamente. Nótese que los tres problemas tienen el mismo índice de coste $V$, difiriendo únicamente en el conjunto de restricciones.

El problema $\mathcal{P}_{\mathbf{conv}(\mathbb{T})}$ es el correspondiente a un sistema lineal con restricciones convexas y, como se vio en el capítulo 2, tiene una solución explícita de la forma:

$$\mathbf{u}_C^{opt}(x) = \mathbf{u}_{Ci}^{opt}(x) = G_i^C x + h_i^C, \quad \text{para } x \in X_i^C \tag{4.21}$$

$$\text{donde } X_i^C = \{x \in \mathbb{R}^n | L_i^C x \leq W_i^C\} \tag{4.22}$$

En esta sección, se detalla cómo puede utilizarse esta solución para obtener la solución al problema $\mathcal{P}_{\mathbb{T}}$. Para ello, en primer lugar se formula la siguiente proposición:

**Proposición 4.8.** *Sea $X_i^C$ una región de la solución explícita de $\mathcal{P}_{\mathbf{conv}(\mathbb{T})}$ para la que la expresión del óptimo es $\mathbf{u}_{Ci}^{opt}(x)$. Este óptimo es factible en $\mathbb{T}$ para todo $x \in X_i^C$ si y solo si*

$$X_i^C \subset X_j$$

*donde $X_j$ es una región de la solución explícita de $\mathcal{P}_{\mathbb{T}}$ con la misma expresión del óptimo, $\mathbf{u}_j^{opt}(x) = \mathbf{u}_{Ci}^{opt}(x)$.* □

**Prueba.** En primer lugar, si $X_i^C \subset X_j$, $x \in X_i^C \Rightarrow x \in X_j$ y por tanto el óptimo es también factible en $\mathbb{T}$. Por otro lado, como $\mathbb{T} \subset \mathbf{conv}(\mathbb{T})$, cualquier solución factible en $\mathcal{P}_{\mathbb{T}}$ lo es también en $\mathcal{P}_{\mathbf{conv}(\mathbb{T})}$. Por tanto, si para un determinado $x \in X_i^C$ el óptimo en $\mathcal{P}_{\mathbf{conv}(\mathbb{T})}$ es factible en $\mathcal{P}_{\mathbb{T}}$, también debe ser el óptimo en $\mathcal{P}_{\mathbb{T}}$, $x \in X_j$. Por tanto, $X_i^C \subset X_j$. ■

A la vista del resultado anterior, se pueden clasificar las regiones de la solución explícita de $\mathcal{P}_{\mathbf{conv}(\mathbb{T})}$, $X_i^C$, en tres grupos dependiendo de su relación con las regiones de la solución explícita a $\mathcal{P}_{\mathbb{T}}$, $X_j$:

**Regiones coincidentes:** Regiones cuya expresión afín del óptimo es factible en $\mathbb{T}$ para cualquier $x \in X_i^C$ y coinciden exactamente con alguna de las regiones solución de $\mathcal{P}_{\mathbb{T}}$, $X_j$:

$$X_i^C = X_j$$

**Regiones subconjunto:** Regiones cuya expresión afín del óptimo es factible en $\mathbb{T}$ para cualquier $x \in X_i^C$ pero no coinciden exactamente con ninguna de las regiones $X_j$, sino que son un subconjunto estricto:

$$X_i^C \subset X_j$$

**Regiones no factibles:** Regiones cuya expresión afín del óptimo no es factible en $\mathbb{T}$ para algunos $x \in X_i^C$. Por aplicación de la proposición 4.8, estas regiones cumplen:

$$\nexists X_j | X_i^C \subseteq X_j$$

**Observación 4.3.** Por definición, dado que $\mathbb{T}$ es un poliedro no convexo, existen algunos valores de $(\mathbf{u}, x)$ que están en el interior de $\mathbf{conv}(\mathbb{T})$ pero no de $\mathbb{T}$, y por tanto la región correspondiente a un óptimo sin ninguna restricción activa ($X_0^C$) es una región no factible. □

A partir de la clasificación anterior, se propone la metodología alternativa para la resolución del problema $\mathcal{P}_{\mathbb{T}}$. La idea es obtener la envolvente convexa de la unión de los diferentes conjuntos de restricciones $T_i$ ($\mathbf{conv}(\mathbb{T})$) y obtener la solución explícita de $\mathcal{P}_{\mathbf{conv}(\mathbb{T})}$. A continuación, se clasifican todas las regiones de dicha solución explícita en los tres grupos descritos anteriormente. Posteriormente, únicamente es necesario analizar en detalle las regiones no factibles, cuya solución no satisface ninguno de los $T_i$. El algoritmo 4.9 resume esta metodología.

**Algoritmo 4.9** Metodología de la envolvente convexa

1. Cálculo de la envolvente convexa de todos los subconjuntos $T_i$.
2. Obtención de la solución explícita del problema $\mathcal{P}_{\mathbf{conv}(\mathbb{T})}$.
3. Clasificación de las regiones de dicha solución:

   **Regiones coincidentes:** Regiones coincidentes con alguna región de la solución explícita de $\mathcal{P}_{\mathbb{T}}$.

   **Regiones subconjunto:** Regiones con solución factible que son subconjunto de alguna región de la solución de $\mathcal{P}_{\mathbb{T}}$.

   **Regiones no factibles:** Regiones que contienen algunos $x$ para los que la solución explícita no es factible en $\mathbb{T}$.
4. División de las regiones no incluidas.
5. Eliminación de soluciones no óptimas en las regiones divididas.
6. Unión de regiones no incluidas que no se han dividido y regiones subconjunto.

### 4.4.1. Cálculo de la envolvente convexa

En primer lugar, es necesario el cálculo de la envolvente convexa del poliedro no convexo que define las restricciones del nuevo problema de optimización, $\mathbb{T}$, mediante alguno de los métodos enumerados en la sección 2.6.1.

En este punto es interesante comentar que, aunque se ha formulado para la envolvente convexa, la proposición 4.8 es de aplicación para cualquier conjunto convexo que contenga a todos los conjuntos $T_i$. Por tanto, en aquellos casos en los que el cálculo de la envolvente sea demasiado costoso, puede substituirse por otro poliedro convexo.

Una primera opción es utilizar la envoltura de los poliedros, $env(T_i)$ (definición 2.12).

Otra posibilidad es definir el *problema convexo*, consistente en substituir los poliedros no convexos en el problema de optimización (3.2) por sus envolventes convexas, antes de formar los conjuntos $T_i$ definidos en $(\mathbf{u}, x)$:

$$\begin{aligned} \mathcal{P}_{N,M}(x): \quad & V_{N,M}^{OPT}(x) := \min \tfrac{1}{2}\mathbf{u}^T H \mathbf{u} + \mathbf{u}^T F x, \\ & \text{sujeto a:} \\ & u_k \in \mathbf{conv}(\bar{\mathbb{U}}) \text{ para } k = 0, \ldots, M-1, \\ & u_k = 0 \text{ para } k = M, \ldots, N-1, \\ & (\Omega_k x + \Gamma_k \mathbf{u}) \in \mathbf{conv}(\bar{\mathbb{X}}) \text{ para } k = 0, \ldots, N, \\ & (\Omega_N x + \Gamma_N \mathbf{u}) \in \mathbf{conv}(\bar{\mathbb{X}}_f) \end{aligned}$$

Al ser todos los conjuntos de restricciones convexos, se tiene un único conjunto de restricciones que también contendrá a todos los $T_i$.

La ventaja de utilizar cualquiera de estas dos alternativas es que su cálculo es considerablemente más sencillo. En el caso de la envoltura porque, como se vio en la sección 2.6.2, su construcción consiste simplemente en eliminar algunas de las restricciones que forman las regiones $T_i$ originales. En el caso del problema convexo, porque las envolventes que es necesario calcular están definidas en una dimensión mucho menor ($m$ y $n$ en lugar de $m \cdot M + n \cdot N$).

No obstante, ambas alternativas presentan una desventaja: $\mathbf{conv}(\mathbb{T})$ es, por definición, el mínimo poliedro convexo que contiene a todos los conjuntos $T_i$. Por tanto, cuando las restricciones son definidas mediante cualquiera de las otras dos alternativas, es probable que aparezcan en la solución explícita un mayor número de regiones no factibles, aumentando el coste de las fases posteriores del algoritmo.

### 4.4.2. Clasificación de las regiones de la solución explícita

El primer paso para la clasificación de las regiones de la solución de $\mathcal{P}_{\mathbf{conv}(\mathbb{T})}$ será comprobar cuáles de ellas tienen una solución factible en $\mathbb{T}$, diferenciando de esta forma las regiones no factibles de las coincidentes y subconjunto. Cómo se ha visto anteriormente, la región sin ninguna restricción activa, $X_0^C$, es por definición no factible, por lo que sólo será necesario comprobar el resto de regiones.

Para ello, en el caso general, en primer lugar se obtiene el complemento de $\mathbb{T}$, es decir, las regiones cuya unión determina la diferencia entre $\mathbb{T}$ y $\mathbf{conv}(\mathbb{T})$:

$$\mathbb{T}^* = \mathbf{conv}(\mathbb{T}) \backslash \mathbb{T} = \bigcup T_j^*$$

A continuación, se toma la solución explícita $\mathbf{u}_{Ci}^{opt}(x)$ para la región que se está analizando y uno de los conjuntos de restricciones del complemento, $T_j^*$:

$$\begin{aligned} \mathbf{u}_{Ci}^{opt}(x) &= G_i^C x + h_i^C \\ T_j^* &:= \left\{ (\mathbf{u}, x) \,\middle|\, \Phi_j^* \mathbf{u} + \Lambda_j^* x \leq \Delta_j^* \right\} \end{aligned}$$

Se puede realizar la comprobación de para qué subconjunto del espacio de estados la acción de control calculada satisface dichas restricciones:

$$\begin{aligned} \Phi_j^* \mathbf{u}_{Ci}^{opt} &\leq \Delta_j^* - \Lambda_j^* x \\ \Phi_j^* \left( G_i^C x + h_i^C \right) &\leq \Delta_j^* - \Lambda_j^* x \\ \left( \Phi_j^* G_i^C + \Lambda_j^* \right) x &\leq \Delta_j^* - \Phi_j^* h_i^C \end{aligned} \tag{4.23}$$

Siempre que exista algún $x \in X_i^C$, región para la que es válida la expresión de $\mathbf{u}_{Ci}^{opt}(x)$, para el que el óptimo $\mathbf{u}_{Ci}^{opt}(x)$ cumpla (4.23), éste no será factible

para las restricciones $\mathbb{T}$, y por tanto la región $X_i^C$ será de tipo no factible. Esto puede determinarse comprobando la factibilidad del siguiente sistema:

$$\begin{cases} L_i^C x \leq W_i^C \\ \left(\Phi_j^* G_i^C + \Lambda_j^*\right) x \leq \Delta_j^* - \Phi_j^* h_i^C \end{cases} \tag{4.24}$$

Si el problema resulta ser consistente, la región $X_i^C$ será no factible.

En caso de que el sistema no sea consistente, será necesario resolver el mismo problema para un nuevo conjunto $T_j^*$. Si para ninguno de los conjuntos $T_j^*$ el problema de factibilidad es consistente, y teniendo en cuenta que por definición el óptimo para la región es factible en $\mathbf{conv}(\mathbb{T})$, puede afirmarse que también es factible en $\mathbb{T}$, y por lo tanto la región es coincidente o subconjunto.

Una vez se han identificado las regiones no factibles, para completar la clasificación de regiones es necesario diferenciar entre las coincidentes y subconjunto, averiguando cuáles de ellas coinciden exactamente con alguna de las regiones de la solución a $\mathcal{P}_{\mathbb{T}}$. Para ello, se formula la siguiente proposición:

**Proposición 4.9.** *Sea $X_i^C$ una región de la solución explícita de $\mathcal{P}_{\mathbf{conv}(\mathbb{T})}$ con un óptimo $\mathbf{u}_{Ci}^{opt}(x)$ y $X_{i^*}^C$ sus regiones adyacentes excepto $X_0^C$. Si ninguna de ellas es una región no factible, entonces*

$$X_i^C = X_j$$

*donde $X_j$ es la región de la solución explícita de $\mathcal{P}_{\mathbb{T}}$ con la misma expresión del óptimo, $\mathbf{u}_j^{opt}(x) = \mathbf{u}_{Ci}^{opt}(x)$.* □

**Prueba.** De la proposición 4.8 y la factibilidad en $\mathbb{T}$ del óptimo, se tiene $X_i^C \subset X_j$, por lo que sólo es necesario probar $X_j \subset X_i^C$.

Para ello, considérese que la región $X_i^C$ está delimitada por las facetas compartidas con sus regiones adyacentes, que son o bien coincidentes o subconjunto, o la región $X_0^C$. Por definición, las regiones coincidentes y subconjunto tienen un óptimo factible en $\mathbb{T}$ y por tanto $X_j$ no puede llegar más allá que estas facetas compartidas (el óptimo cambiaría al válido en $X_{i^*}^C$).

Por último, si $X_i^C$ es adyacente a $X_0^C$, el argumento anterior no es válido, puesto que es una región no factible. Sin embargo, el óptimo en esta región es el mismo que para el problema sin restricciones $\mathbf{u}^{opt}(x) = -H^{-1}Fx$. Este óptimo es válido para cualquier $x \in X_0^C$ y, en particular, en la faceta $f$ que la región comparte con $X_i^C$. Como ésta es una región coincidente o subconjunto, $\mathbf{u}^{opt}(x) \in \mathbb{T}$, es decir, existe algún $T_i$ para el que el óptimo es factible, satisfaciendo

$$\left(-\Phi_i H^{-1} F + \Lambda_i\right) x \leq \Delta_i$$

que es una de las regiones solución a $\mathcal{P}_{T_i}$. De esta forma, se puede aplicar el mismo argumento utilizado para las regiones coincidentes y subconjunto, y $X_j$ no puede pasar de $f$.

Como $X_j$ está delimitada por las facetas de $X_i^C$, se deduce $X_j \subset X_i^C$. ■

La proposición anterior puede utilizarse para elaborar un procedimiento para distinguir las regiones $X_i^C$ coincidentes y subconjunto. Para ello, basta con identificar todas las regiones adyacentes y comprobar si alguna de ellas, aparte de la región $X_0^C$, es no factible. En caso afirmativo, la región en cuestión es subconjunto, $X_i^C \subset X_j$. Si no, es coincidente $X_i^C = X_j$.

El algoritmo 4.10 resume el procedimiento para la clasificación de regiones.

**Algoritmo 4.10** Clasificación de regiones de la solución explícita de $\mathcal{P}_{\mathbf{conv}(\mathbb{T})}$.

1. Obtener el complemento del conjunto de restricciones $\mathbb{T}$ en $\mathbf{conv}(\mathbb{T})$:

$$\mathbb{T}^* = \mathbf{conv}(\mathbb{T}) \backslash \mathbb{T} = \bigcup T_j^*$$

2. Para cada región $X_i^C$, comprobar la factibilidad en $\mathbb{T}^*$ de la solución $\mathbf{u}_{Ci}^{opt}$.
   - Para cada conjunto $T_j^*$:
     *a*) Comprobar factibilidad de (4.23).
     *b*) Si el sistema es factible para algún $T_j^*$, $X_i^C$ es una región no factible.
   - Si $\nexists T_j^*$ para el que (4.23) sea factible, $X_i^C$ es coincidente o subconjunto.
3. Para cada $X_i^C$ que no sea no factible:
   - Identificar sus regiones adyacentes $X_{i^*}^C$.
   - Comprobar si las $X_{i^*}^C \neq X_0^C$ son no factibles:
     *a*) Si alguna de ellas es no factible, $X_i^C$ es una región subconjunto ($X_i^C \subset X_j$).
     *b*) En caso contrario, $X_i^C$ es una región coincidente ($X_i^C = X_j$).

### 4.4.3. División de las regiones no factibles

Tras la clasificación de regiones realizada se tiene que, para las regiones coincidentes y subconjunto, se conoce la expresión del óptimo solución al problema $\mathcal{P}_{\mathbb{T}}$ (porque es la misma que para el problema $\mathcal{P}_{\mathbf{conv}(\mathbb{T})}$), mientras que para las no factibles es desconocido. En está sección, se introduce un procedimiento para calcular este óptimo.

Para ello, la idea es seguir un razonamiento similar al llevado a cabo mediante el algoritmo de intersección, división y unión, si bien en este caso sólo será necesario aplicarlo en el interior de las regiones no factibles, en lugar de en todo el espacio de estados.

Dada una región $X_j$, el procedimiento se inicia tomando uno de los conjuntos de restricciones $T_i$ y dividiendo $X_j$ en subregiones $X_{ji_1}$ en función de la solución al problema $\mathcal{P}_{T_i}$. De esta forma, se obtendrán regiones con diferente expresión explícita de dicha solución, caracterizada por conjuntos diferentes de restricciones activas[1]. Si esto se hace para todas las regiones no factibles, puede ocurrir que para diferentes $X_{ji_1}$ se tenga una misma expresión afín de la solución explícita. Por tanto, puede comprobarse, mediante los procedimientos introducidos en la sección 2.6.2, si es posible sustituirlas por un número menor de regiones convexas.
Una vez se completa este primer paso, todas las regiones no factibles están divididas en subregiones con diferentes óptimos solución del problema $\mathcal{P}_{T_i}$ (o, en su caso, sin ningún óptimo). En este punto, puede considerarse un nuevo conjunto de restricciones, $T_{i+1}$. Siguiendo la misma filosofía, las regiones $X_{ji_1}$ se pueden dividir en subregiones dependiendo de la solución óptima a $\mathcal{P}_{T_{i+1}}$ que presentan. De esta forma, se obtiene un nuevo conjunto de regiones, $X_{ji_1i_2}$, para las que las soluciones a los problemas $\mathcal{P}_{T_i}$ y $\mathcal{P}_{T_{i+1}}$ tienen diferentes expresión explícita. De nuevo, subregiones que vienen de dividir regiones $X_{ji_1}$ con la misma solución pero unión no convexa, pueden tener la misma pareja de soluciones explícitas de los dos problemas. Si estas regiones tienen unión convexa, pueden reemplazarse por un único poliedro.

Siguiendo este enfoque para los $\gamma$ conjuntos $T_i$ diferentes, finalmente se tendrá todo espacio cubierto por las regiones no factibles dividido en un cierto número de regiones $X_{ji_1i_2\ldots i_\gamma}$, cada una de ellas con $\gamma$ soluciones explícitas posibles, una para cada conjunto de restricciones diferente.

Si se analiza la metodología descrita, ésta se basa fundamentalmente en resolver repetidas veces un mismo problema: dada una determinada región, obtener todas las regiones activas posibles en su interior para un determinado conjunto de restricciones. La solución a este problema, que también aparece al plantear el control predictivo explícito de sistemas con restricciones lineales y convexas, ya ha sido descrita en el capítulo 2, donde se vio la existencia de diferentes algoritmos con una estructura común (algoritmo 2.1). El algoritmo 4.11 se basa en dicha estructura para la obtención de todas las divisiones en regiones lineales necesarias.

Las únicas operaciones del algoritmo 4.11 que conllevan cierta complejidad son la búsqueda de un punto factible y caracterización de la región crítica a la que pertenece, la búsqueda de todas las regiones adyacentes a una dada a través de una determinada faceta y la minimización del número de elementos que definen un poliedro no convexo. El primer problema aparece igualmente en la caracterización de la solución explícita para sistemas con restricciones convexas, y se resuelve mediante un LP (que busca el punto factible) y un QP (que indica qué restricciones se encuentran activas para dicho punto). En cuanto al segundo problema, se trató también en el capítulo 2 (algoritmo 2.2). Por último, la minimización del número de elementos de un poliedro no convexo

[1] Puede ocurrir que para algunos puntos de la región $X_j$ no exista solución al problema $\mathcal{P}_{T_i}$. Estos puntos también formarán regiones $X_{ji_1}$, aunque sin ninguna solución asociada.

**Algoritmo 4.11** Exploración y división de regiones no factibles

1. Inicializar conjunto de restricciones no exploradas: $\mathbb{T} = \{T_1, \ldots, T_\gamma\}$,
2. Inicializar conjunto de regiones a dividir con todas las regiones no factibles de la solución a $\mathcal{P}_{\mathbf{conv}(\mathbb{T})}$: $\mathbb{X} = \{X_j\}$.
3. Mientras $\mathbb{T} \neq \emptyset$, tomar $T_i \in \mathbb{T}$:
   - *a*) Inicializar conjunto de nuevas regiones por dividir: $\mathbb{Y} = \emptyset$.
   - *b*) Tomar $X \in \mathbb{X}$:
     1) Buscar un punto $x$ factible en $X$ y caracterizar su región crítica para el problema $\mathcal{P}_{T_i}$: $X_1$.
     2) Inicializar el conjunto de regiones conocidas $\mathbb{R} := \{X_1\}$ y el de regiones inexploradas $\mathbb{U} := \{X_1\}$.
     3) Mientras $\mathbb{U} \neq \emptyset$:
        - $a'$ Elegir un elemento $U \in \mathbb{U}$ y hacer $\mathbb{U} = \mathbb{U} \backslash U$.
        - $b'$ Buscar facetas $f$ de $U$ que pasen por $X$.
        - $c'$ Para todas las facetas $f$:
          - Aplicar algoritmo 2.2 para buscar las regiones de la solución a $\mathcal{P}_{T_i}$ adyacentes a $U$ a través de $f$: $S_i$.
          - $\mathbb{U} = \mathbb{U} \cup \{S_i \backslash \mathbb{R}\}$
          - $\mathbb{R} = \mathbb{R} \cup S_i$
     4) $\mathbb{Y} = \mathbb{Y} \cup \mathbb{R}$.
     5) $\mathbb{X} = \mathbb{X} \backslash X$.
   - *c*) Si $\mathbb{X} \neq \emptyset$: volver a 3.b.
   - *d*) Si $\mathbb{X} = \emptyset$:
     - Buscar regiones de $\mathbb{Y}$ con las mismas soluciones posibles, aplicar algoritmo de unión de regiones 4.4 cuando sea posible y actualizar $\mathbb{Y}$.
     - $\mathbb{X} = \mathbb{Y}$, $\mathbb{T} = \mathbb{T} \backslash T_i$, volver a 3.

se discutió en la sección 2.6.2.

### 4.4.4. Eliminación de soluciones no óptimas

Una vez alcanzado este punto, se tiene un conjunto de regiones $X_{ji_1i_2\ldots i_\gamma}$ (obtenidas de las sucesivas divisiones de las regiones no factibles) con $\gamma$ soluciones afines diferentes posibles. Aunque sería posible finalizar el algoritmo en este punto y comparar todos los costes en el algoritmo en línea, es conveniente comprobar fuera de línea cuáles de las $\gamma$ posibles soluciones son realmente factibles en estas regiones lineales.

Exactamente este mismo problema ha sido resuelto en secciones anteriores para el algoritmo de intersección, división y unión (algoritmo 4.7), eliminado mediante máximos y mínimos algunas de las soluciones y, para el resto, comprobando mediante optimización SOS la factibilidad del siguiente sistema:

$$\begin{cases} L_r x \leq W_r \\ V_k^{OPT} \leq V_l^{OPT} \quad l = \{1\ldots(j+1)\}\backslash\{k\} \end{cases}$$

El algoritmo 4.12 resume ese mismo procedimiento para el problema actual.

**Algoritmo 4.12** Eliminación de las soluciones no óptimas.

1. Para cada región $X_{ji_1i_2\ldots i_\gamma}$ obtener el conjunto de $s \leq \gamma$ soluciones posibles.
2. Inicializar el conjunto de soluciones no óptimas: $\mathcal{NO} = \emptyset$.
3. Obtener los máximos y mínimos de cada índice de coste, $\max(V_{iN}^{OPT})$, $\min(V_{iN}^{OPT})$.
4. Para $k = 1$ hasta $s$:
   *a*) Si $\exists l \in \{1\ldots s\}\backslash\{k\} : \min(V_k^{OPT}) \geq \max(V_l^{OPT})$, entonces $\mathcal{NO}(X_r) = \mathcal{NO}(X_r) \cup k$.
5. Obtener el conjunto de soluciones óptimas: $\mathcal{O}(X_r) = \{1\ldots s\}\backslash\mathcal{NO}(X_r)$.
6. Para $k \in \mathcal{O}(X_r)$:
   *a*) Resolver el problema SOS:
   $$\begin{cases} L_r x \leq W_r \\ V_k^{OPT} \leq V_l^{OPT} \quad l = \{1\ldots s\}\backslash\{\mathcal{NO} \cup k\} \end{cases}$$
   *b*) Si el problema no es factible, $\mathcal{NO} = \mathcal{NO} \cup k$.
7. Actualizar el conjunto de soluciones óptimas: $\mathcal{O} = \mathcal{O}\backslash\mathcal{NO}$

### 4.4.5. Unión de regiones con las mismas soluciones

Una vez la división de regiones no factibles se ha realizado, éstas son substituidas por un determinado número de regiones, cada una de ellas con un conjunto de soluciones óptimas posibles diferentes. Posteriormente, para cada una de estas nuevas regiones se realiza la eliminación de soluciones no óptimas. En este punto, puede ocurrir que para algunas regiones que fueron divididas por diferir en algunas de sus soluciones posibles, éstas soluciones sean eliminadas por ser no óptimas. Por ejemplo, supóngase una región $X_{ji_1}$ con una solución óptima dada para el conjunto de restricciones $T_1$ que se subdivide en $X_{ji_1i_2}$, $X_{ji_1i_3}$ y $X_{ji_1i_4}$, porque el óptimo para el problema con restricciones $T_2$ es diferente. Si, tras resolver los problemas SOS correspondientes se comprueba que en dos de las regiones, $X_{ji_1i_2}$ y $X_{ji_1i_3}$, la única región óptima es la correspondiente al conjunto de restricciones $T_1$, estas regiones pueden unirse.

Por otra parte, cuando la eliminación de soluciones no óptimas en una región deja una única solución óptima posible, puede ocurrir que ésta coincida con la solución de una región subconjunto adyacente, y por tanto también son susceptibles de ser unidas.

En ambos casos, es posible aplicar los algoritmos de minimización del número de elementos en un poliedro no convexo introducidos en la sección 2.6.2.

**Ejemplo 4.9.** *Se va a resolver, mediante la metodología de la envolvente convexa, el mismo problema resuelto mediante la metodología de intersección, división y unión en el ejemplo 4.6. Como se vio en dicho ejemplo, el problema de optimización a resolver, $\mathcal{P}_{\mathbb{T}}$, puede expresarse en la forma* (3.5) *con unas matrices $H$ y $F$:*

$$H = \begin{bmatrix} 1{,}980 & 1{,}988 \\ 1{,}988 & 11{,}49 \end{bmatrix} \quad F = \begin{bmatrix} 0{,}5942 & 1{,}988 \\ 1{,}206 & 10{,}49 \end{bmatrix}$$

*Dicho problema tiene unas restricciones de la forma $\mathbb{T} = T_1 \cup T_2$ con unas $T_i = \mathbb{U}_i$ como las representadas en la figura 4.18.*

*El procedimiento para la resolución del problema es el resumido en el algoritmo 4.9 y se inicia obteniendo la envolvente convexa de la unión de los conjuntos de restricciones $\mathbb{T}$ (figura 4.19) y obteniendo la solución explícita del problema $\mathcal{P}_{\mathbf{conv}(\mathbb{T})}$ (figura 4.20).*

*A continuación, se realiza la clasificación de las regiones de la partición, tal como se describe en el algoritmo 4.10. Para ello, en primer lugar se obtiene $\mathbb{T}^* = \mathbf{conv}(\mathbb{T})\backslash\mathbb{T}$ y se comprueba, mediante 11 LPs (uno para cada región de la solución de $\mathcal{P}_{\mathbf{conv}(\mathbb{T})}$), que la únicas regiones no factibles son $\mathbb{X}_1$ y $\mathbb{X}_3$.*

*Buscando las que son adyacentes a éstas, y teniendo en cuenta que $\mathbb{X}_0^C = \mathbb{X}_1$ puede finalizarse la clasificación de las regiones:*

**Coincidentes:** $\mathbb{X}_2$, $\mathbb{X}_4$, $\mathbb{X}_5$, $\mathbb{X}_6$, $\mathbb{X}_7$, $\mathbb{X}_8$, $\mathbb{X}_{11}$.

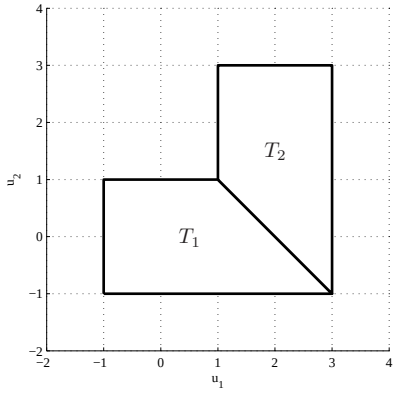


***Figura 4.18:*** Restricciones $\mathcal{P}_{\mathbb{T}}$.

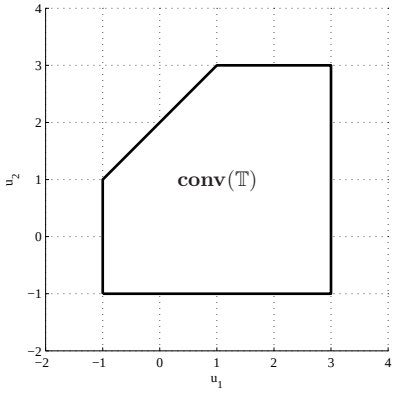


***Figura 4.19:*** Restricciones $\mathcal{P}_{\mathbf{conv}(\mathbb{T})}$

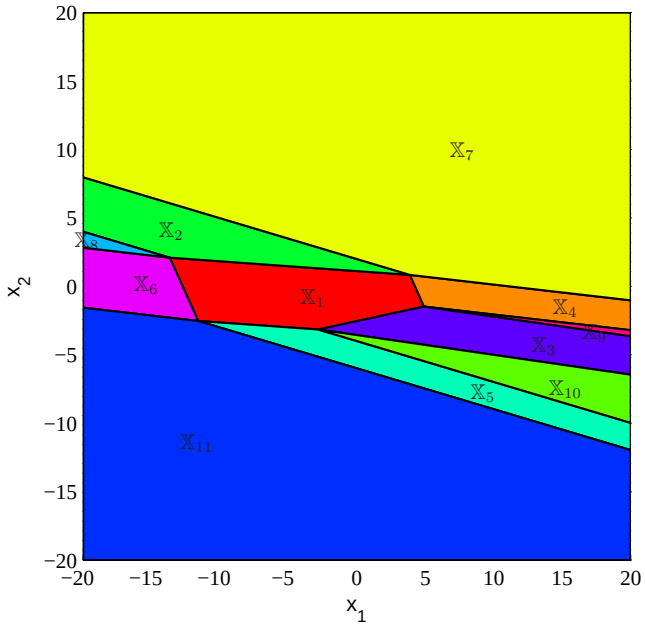


***Figura 4.20:*** Solución $\mathcal{P}_{\mathbf{conv}(\mathbb{T})}$.Partición del estado.

**Subconjunto:** $\mathbb{X}_9$, $\mathbb{X}_{10}$.

**No factibles:** $\mathbb{X}_1$, $\mathbb{X}_3$.

*Una vez obtenida la clasificación de regiones, se aplica al algoritmo 4.11 de exploración y división a las regiones $\mathbb{X}_1$ y $\mathbb{X}_3$. Para ello, inicialmente se busca un punto dentro de $\mathbb{X}_1$, y se caracteriza la región crítica que lo contiene para el problema $\mathcal{P}_{T_1}$ ($\mathbb{X}_{11}$ en la figura 4.21). A partir de las facetas de ésta que no se encuentran en la frontera de $\mathbb{X}_1$, se obtienen las otras tres regiones que se observan en la figura. Del mismo modo, partiendo de $\mathbb{X}_3$ se encuentra inicialmente $\mathbb{X}_{31}$ y a partir de la faceta correspondiente, las otras dos regiones.*

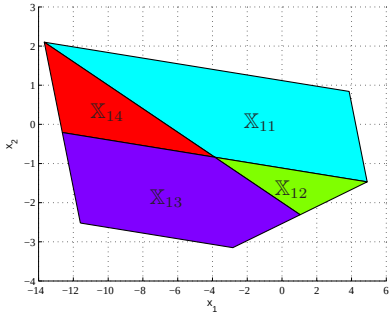


***Figura 4.21:*** División de $\mathbb{X}_1$ según $\mathcal{P}_{T_1}$

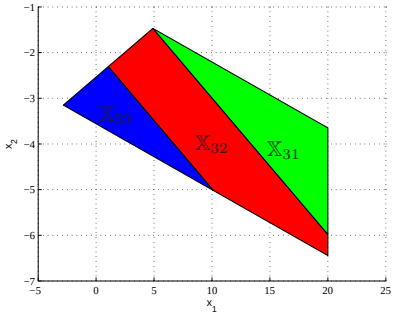


***Figura 4.22:*** División de $\mathbb{X}_3$ según $\mathcal{P}_{T_1}$

*Dado que en $\mathbb{X}_{12}$ y $\mathbb{X}_{32}$ por un lado, y en $\mathbb{X}_{13}$ y $\mathbb{X}_{33}$ por el otro, se tienen respectivamente las mismas soluciones afines a $\mathcal{P}_{T_1}$, se les aplica a cada pareja de regiones el algoritmo 4.4 de unión de regiones. El resultado en el primer caso es una nueva región convexa que puede reemplazar a las anteriores, pero en el segundo caso la unión formada no es convexa (figura 4.23), por lo que habrá que conservar ambas regiones para las iteraciones siguientes. La figura 4.23 muestras las subregiones finales de $\mathbb{X}_1$ y $\mathbb{X}_3$ con arreglo a la solución explícita de $\mathcal{P}_{T_1}$.*

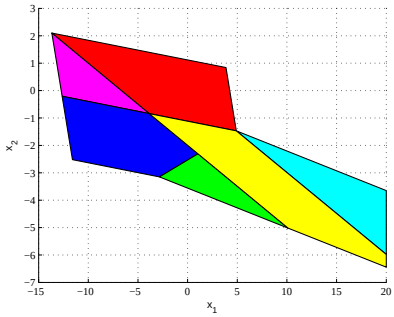


***Figura 4.23:*** Uniones tras $\mathcal{P}_{T_1}$

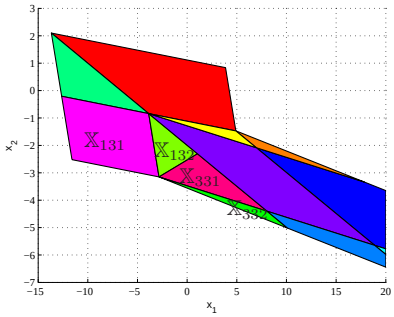


***Figura 4.24:*** División según $\mathcal{P}_{T_2}$

*A continuación, se aplica el mismo procedimiento de exploración en el interior de todas las regiones de la figura 4.23 considerando la caracterización de las regiones críticas de la solución de $\mathcal{P}_{T_2}$. Con ello, se obtienen las regiones de la figura 4.24. De todas ellas, sólo serán susceptibles de unión aquellas en las que todas las soluciones posibles coincidan. Esto sólo ocurre para las regiones $\mathbb{X}_{132}$ y $\mathbb{X}_{331}$. Siendo la unión de estas regiones convexa, pueden reemplazarse por un única (figura 4.25).*

*Puesto que no hay más conjuntos de restricciones posibles $T_j$, la fase de exploración de regiones finaliza en este punto. Por tanto, de todas las regiones en el espacio de estados, en este punto sólo se desconoce que posible solución es la mejor en las 11 regiones de la figura 4.25. Es en estas regiones en las que será necesario aplicar el algoritmo 4.12. Dado que en cada una de las regiones sólo hay dos posibles soluciones, basta con resolver un problema SOS para cada una de ellas. Los resultados obtenidos muestran que para sólo dos de las 11 regiones las dos posibles soluciones son óptimas (estas regiones se muestran divididas por la curva cuadrática correspondiente en la figura 4.26).*

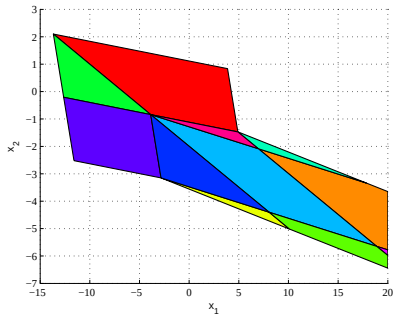


***Figura 4.25:*** Unión tras $\mathcal{P}_{T_2}$

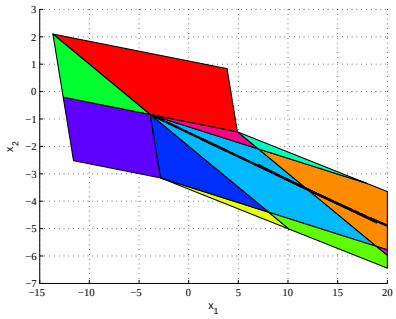


***Figura 4.26:*** Eliminación de soluciones

*Para finalizar, se sustituyen todas las regiones de la figura 4.26 por $\mathbb{X}_1$ y $\mathbb{X}_3$ en la partición solución a $\mathcal{P}_{\mathbf{conv}(\mathbb{T})}$ (figura 4.20) y se aplica el algoritmo de unión de regiones a las 9 regiones no divididas y todas las regiones subconjunto ($\mathbb{X}_9$ y $\mathbb{X}_{10}$). El resultado final obtenido se muestra en la figura 4.27.*

*Puede comprobarse que, como era de esperar, el resultado coincide exactamente con la figura 4.11, que muestra la partición del estado obtenida al aplicar el método de intersección, división y unión al mismo problema. La expresión del controlador también coincide con* (4.13). ■

**Ejemplo 4.10.** *Se considera el mismo sistema lineal de los ejemplos 4.6 y 4.9:*

$$G(s) = \begin{bmatrix} \dfrac{1}{s+0{,}5} & \dfrac{1}{s} \end{bmatrix}$$

*discretizado con un período de $T_s = 1$. El sistema está sujeto a las mismas restricciones sobre las acciones de control $u(t) \in \bar{\mathbb{U}} = \mathbb{U}_1 \cup \mathbb{U}_2$ pero, además, se*

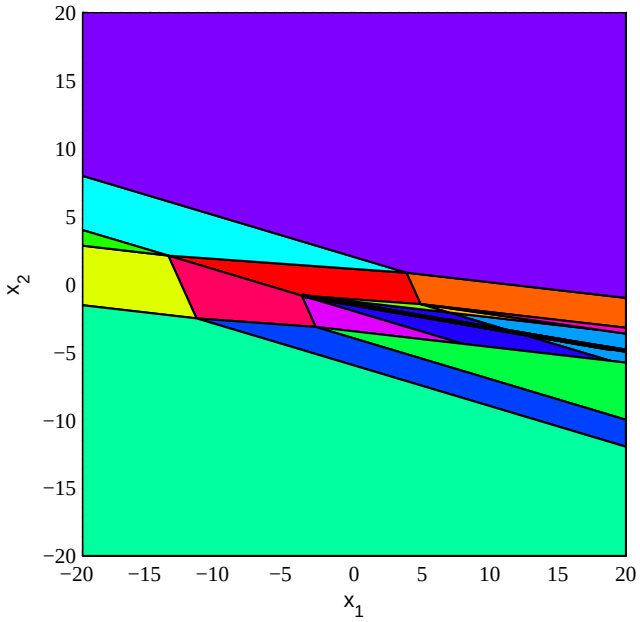


***Figura 4.27:*** Solución final. Partición del estado.

*añaden restricciones sobre los estados,* $-40 \leq x(t) \leq 40$.

$$\mathbb{U}_1 := \left\{ \begin{bmatrix} -1 & 0 \\ 0 & 1 \\ 1 & 1 \\ 0 & -1 \end{bmatrix} u \leq \begin{bmatrix} 1 \\ 1 \\ 2 \\ 1 \end{bmatrix} \right\} \quad \mathbb{U}_2 := \left\{ \begin{bmatrix} -1 & -1 \\ -1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} u \leq \begin{bmatrix} -2 \\ -1 \\ 3 \\ 3 \end{bmatrix} \right\}$$

*Los parámetros de diseño del controlador predictivo son* $N = 1$, $Q = C^T C$ *y* $R = I_{2\times 2}$.

*Con objeto de garantizar la estabilidad del sistema en bucle cerrado, se añade un conjunto de restricciones terminal* $\mathbb{X}_f$[2]. *Dicho conjunto terminal es un poliedro no convexo formado por dos elementos, por lo que existen 4 conjuntos diferentes de restricciones* $T_i$. *Calculando la envolvente convexa y definiendo el problema con las nuevas restricciones, se obtiene una solución explícita definida en 20 regiones (figura 4.28).*

*A continuación, se aplican los algoritmos 4.10 y 4.11, obteniéndose la clasificación de regiones y la división de las no factibles. Tras estas operaciones, se obtiene la partición del espacio de estados mostrada en la figura 4.29, con 141 regiones.*

*En este punto, se aplica el algoritmo 4.12. Para ello, es necesario resolver 212 problemas SOS de la forma* (4.15)*, con los que se obtiene que un total de*

[2]En el capítulo 6 se proponen algoritmos para la obtención de este conjunto.

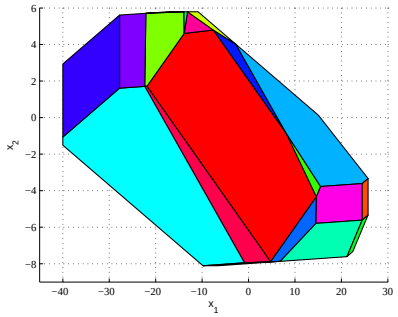


***Figura 4.28:*** Partición para $\mathcal{P}_{\mathbf{conv}(\mathbb{T})}$.

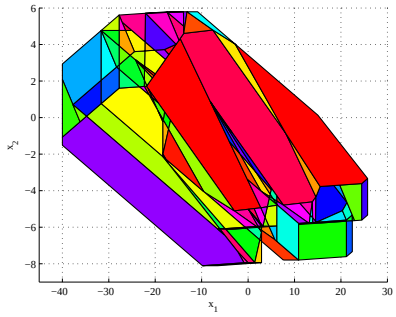


***Figura 4.29:*** Partición antes de eliminación de soluciones.

*163 soluciones no son óptimas, y por tanto pueden eliminarse. Esto deja una partición de 10 regiones con 2 posibles soluciones y 131 regiones con una única solución. Aplicando por última vez el algoritmo de unión a las regiones con la misma solución, se obtiene la solución explícita final. La partición del espacio de estados para esta solución se muestra en la figura 4.30, siendo el número total de regiones de 53 (10 de ellas con 2 soluciones diferentes).* ■

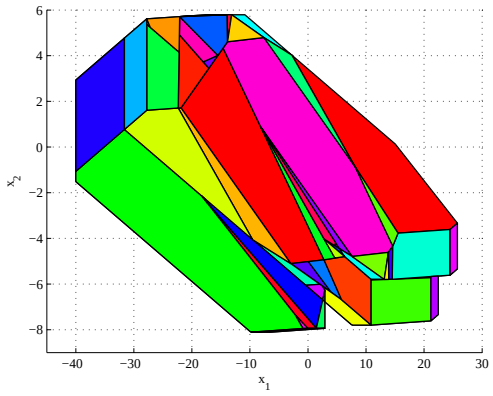


***Figura 4.30:*** Partición poliédrica final.

### 4.4.6. Análisis del coste computacional del algoritmo

Como se ha visto en la sección anterior, el algoritmo de la envolvente convexa para la obtención de la solución explícita del problema (3.3) obtiene la misma solución que el algoritmo de intersección, división y unión. Por ello, el criterio fundamental para decidir utilizar uno u otro algoritmo es el del coste computacional de cada uno de ellos. El coste del algoritmo de intersección, división y unión se analiza en el anexo A, con dos variantes diferentes (algoritmo

simultáneo y progresivo). Para completar el análisis, queda por tanto obtener una estimación del coste computacional del algoritmo de la envolvente convexa.

Al igual que ocurre con el otro algoritmo posible, el cálculo detallado del coste en términos de operaciones elementales resulta demasiado complejo. No obstante, a efectos de comparación entre algoritmos, es posible introducir una serie de simplificaciones:

- *Se medirá el coste en términos de número de los diferentes problemas de optimización a resolver:* Los problemas de optimización son las operaciones más abundantes y costosas en los diferentes algoritmos planteados. Por ello, y aunque no tienen un coste constante, si en uno de los algoritmos es necesario resolver muchos menos problemas de optimización de los diferentes tipos éste será con toda probabilidad el menos costoso computacionalmente.

- *Se obviará la unión de regiones:* Tal y como se ha visto, ésta no requiere la resolución de ningún problema de optimización adicional a los resueltos en pasos anteriores del algoritmo.

- *Para el coste del cálculo de la envolvente convexa, se supondrá que los $\gamma$ conjuntos de restricciones se encuentran en posición general:* Como se ha visto en 2.6.1, se desconoce cuál es en general el mejor algoritmo para el cálculo de la envolvente convexa de un determinado conjunto. No obstante, cuando varios politopos se encuentran en posición general, en [FLL01] se propone un algoritmo que calcula la envolvente convexa de todos ellos y que da una expresión del coste que podemos utilizar como aproximación para la comparación de algoritmos.

- *Se supondrá que el algoritmo de exploración y división de las regiones no factibles es equivalente a la intersección, división y unión en esas regiones:* Aunque el procedimiento que se ha propuesto no es exactamente ese, el problema también podría resolverse obteniendo las particiones correspondientes a cada conjunto de restricciones, comprobando qué regiones intersectan dentro de las regiones no factibles y aplicando la intersección, división y unión de éstas.

Con estas consideraciones, vamos a analizar el coste de los diferentes pasos del algoritmo 4.9.

#### Cálculo de la envolvente convexa

Considerando los $\gamma$ politopos que describen los diferentes conjuntos de restricciones en posición general, el algoritmo propuesto en [FLL01] tiene un coste:

$$O(f_1 d\bar{r} LP(\bar{r}, d) + f_2(\bar{r}d^3 + LP(r+\gamma, \gamma d) + log(f_1)))$$

donde:

- $f_1$ y $f_2$ representan respectivamente el número de facetas y crestas (caras de dimensión $d$-2) que tiene el politopo solución.
- $d = Mm + n$ es la dimensión del espacio en el que se encuentra definida la envolvente convexa.
- $r = r_1 + \dots r_\gamma$, $\bar{r} = max\{r_1, \dots, r_\gamma\}$ siendo $r_i$ el número de desigualdades que define cada uno de los politopos de restricciones $T_i$.
- $LP(i, j)$ es el coste de resolver un problema de programación lineal con $i$ restricciones y $j$ variables.

Despreciando el coste de las operaciones diferentes a los problemas de optimización, y las diferencias en LPs con diferente número de variables y/o restricciones, se tiene un coste del cálculo de la envolvente convexa:

$$\#(LPs) = f_1(Mm + n)\bar{r} + f_2$$

#### Obtención de la solución explícita

Como se vio en la sección 2.3, existen varios algoritmos propuestos por diferentes autores para la obtención de la solución explícita de un problema con restricciones lineales tal como el (4.20), variando el coste computacional dependiendo de cuál de ellos se emplee. No obstante, teniendo en cuenta que como se ha comentado sólo se considera como costoso la resolución de problemas de optimización, podemos resumir las operaciones que tienen interés en el algoritmo de la siguiente forma: para cada región, resolver un LP que busque un punto factible, a continuación un QP para averiguar qué restricciones se encuentran activas, y posteriormente varios LPs para comprobar cuáles de las desigualdades provenientes de las condiciones KKT (2.31a) y (2.31b) son redundantes. El número de LPs que hay que resolver para esta última comprobación dependerá de cuántas desigualdades haya, que dependerá a su vez de cuantas desigualdades definan la envolvente convexa ($f_1$). De esta forma, el número de problemas a resolver es:

$$\#(LPs) = N_{CH}(k + 1) \quad \#(QPs) = N_{CH}$$

donde:

- $N_{CH}$ es el número de regiones en la partición solución a (4.20).
- $k$ es una constante dependiente de $f_1$, y por tanto de cada problema en cuestión.

#### Clasificación de las regiones de la partición anterior

Una vez se tiene la solución explícita anterior, se comprueba cuáles de las $N_{CH}$ son no factibles, es decir, tienen una solución no factible en $\mathbb{T}$. Para ello,

se sigue el algoritmo 4.10 para obtener el complemento de $\mathbb{T}$ en $\mathbf{conv}(\mathbb{T})$, obteniéndose $\gamma^*$ conjuntos $T_i^*$. Para cada región, es necesario resolver un problema de factibilidad (LP) con cada uno de los conjuntos $T_i^*$. El número de problemas LP a resolver es por tanto:

$$\#(LPs) = \gamma^* N_{CH}$$

#### División de regiones no factibles y eliminación de soluciones no óptimas

Como se ha comentado en la enumeración de simplificaciones, para calcular el coste de esta parte del algoritmo, supondremos que se ejecuta un procedimiento diferente al propuesto en el algoritmo 4.11. En su lugar, se toma la primera de las particiones del espacio de estado obtenida con uno de los conjuntos de restricciones originales (por ejemplo $T_1$) y se intersecta con las regiones no factibles existentes. Esto requiere la resolución de $N_{t3}N_1$ problemas de programación lineal. De las $N_1$ regiones de la partición, sólo tendrán intersección con alguna de las regiones no factibles un subconjunto de regiones $N_1'$:

$$N_1' = \delta N_1$$

Dado que para las regiones coincidentes y subconjunto ya se conoce la solución óptima del problema $\mathcal{P}_{\mathbb{T}}$, las $(N_1 - N_1')$ regiones de la solución a $\mathcal{P}_{T_1}$ que no intersectan con alguna de las no factibles, pueden ser ignoradas en el análisis posterior. A continuación, se sustituye esa primera partición por las $N_1'$ regiones que sí tienen intersección no vacía (que no tienen por qué formar una partición). Desde este punto, se aplica el algoritmo de intersección, división y unión (en su variante simultánea o progresiva) tal y como se ha descrito en las secciones anteriores. La expresión del coste de esta parte del algoritmo será, por tanto, la misma que se obtuvo ((A.1) LPs y (A.2) SOS en el caso simultáneo y (A.10) LPs y (A.11) en el progresivo) pero teniendo en cuenta que no se trabaja con las $N_1$ regiones de la primera partición, sino con $N_1'$. Si se observa las expresiones anteriores de coste, y teniendo en cuenta que en todas ellas $N_1$ puede sacarse factor común, se tiene:

$$\#(LPs) = \frac{N_1'}{N_1} C_{LP}^{div} = \delta C_{LP}^{div}$$
$$\#(SOS) = \frac{N_1'}{N_1} C_{SOS}^{div} = \delta C_{SOS}^{div}$$

donde $C_{LP}^{div}$ y $C_{SOS}^{div}$ son respectivamente el número de LPs y SOS a resolver en el algoritmo de intersección, división y unión (sea éste el algoritmo simultáneo o progresivo).

#### Comparación con el algoritmo de intersección, división y unión

Teniendo en cuenta todas las operaciones anteriores, el algoritmo de envolvente convexa requiere resolver el siguiente número de problemas de optimiza-

ción::

$$\begin{aligned} C_{LP}^{CH} &= f_1(Mm+n)\bar{r} + f_2 + N_{CH}\,(k+1+\gamma^*) + N_{t3}N_1 + \delta C_{LP}^{div} \\ C_{QP}^{CH} &= N_{CH} \\ C_{SOS}^{CH} &= \delta C_{SOS}^{div} \end{aligned}$$

En primer lugar, se comparará la diferencia en el número de problemas SOS a realizar. Es sencillo comprobar que $C_{SOS}^{CH} \leq \delta C_{SOS}^{div}$ puesto que, por definición, $\delta < 1$ (las $N_1'$ regiones que tienen intersección no nula con alguna región no factible son un subconjunto de las $N_1$ de la partición).
En cuánto al número de QPs, para el algoritmo de la envolvente convexa siempre habrá que resolver $N_{CH}$ más que para el algoritmo de intersección, división y unión.
Por último, la diferencia en número de LPs dependerá de diversos parámetros:

$$\begin{aligned} C_{LP}^{CH} - C_{LP}^{div} &= f_1(Mm+n)\bar{r} + f_2 + N_{CH}\,(k+1+\gamma^*) + N_{t3}N_1 + \\ &+ \delta C_{LP}^{div} - C_{LP}^{div} = f_1(Mm+n)\bar{r} + f_2 + N_{CH}\,(k+1+\gamma^*) + \\ &+ N_{t3}N_1 - (1-\delta)C_{LP}^{div} \end{aligned}$$

La diferencia anterior siempre tiene un término fijo positivo independiente de $\delta$ (correspondiente fundamentalmente al cálculo de la envolvente convexa, la obtención de la partición de la solución y la clasificación de las regiones de dicha partición) y un término negativo que es mayor cuánto más pequeño es $\delta$. Dependiendo del problema en cuestión, dicha diferencia puede ser positiva o negativa.

Ante los resultados obtenidos, y siempre que no haya más datos de inicio, parece recomendable utilizar el algoritmo de la envolvente convexa. Esto es así porque la resolución de los problemas SOS es bastante más costosa que la de los LPs o QPs (del orden de 10 veces más lenta). Por ello, un algoritmo que reduzca el número de problemas SOS, será muy probablemente menos costoso.

Es interesante analizar también la influencia en los costes anteriores del número de regiones de restricciones $T_i$ posibles, $\gamma$. Si se revisan las expresiones (A.1), (A.2), (A.10),(A.11), puede comprobarse que, tanto para el algoritmo progresivo como el simultáneo, el número de problemas LP y SOS a resolver es del orden $O(\prod^{\gamma} N_i)$.
Esto significa que, a medida que crece $\gamma$, $C_{SOS}^{div}$ crece de manera geométrica. De igual manera, $C_{SOS}^{div} - C_{SOS}^{CH} = (1-\delta)C_{SOS}^{div}$ crecerá geométricamente por lo que, a medida que aumenta el número de particiones, en términos de problemas SOS es cada vez más favorable el algoritmo de la envolvente convexa.
En cuanto a la diferencia de problemas QP, está permanece constante con la variación de $\gamma$.
Por último, la diferencia entre el número de problemas LP a resolver tiene un único término que depende de $\gamma$: $(1-\delta)C_{LP}^{div}$. Dicho término, que afecta negativamente, crece de manera geométrica con $\gamma$. Por tanto, a medida que crece $\gamma$ el número de problemas LP también es más favorable al algoritmo de la envolvente convexa.

## 4.5. Conclusiones

En este capítulo se han introducido dos metodologías diferentes para la obtención de la solución explícita al problema de optimización cuadrática con restricciones poliédricas no convexas.

La primera de estas metodologías se basa en formular subproblemas con restricciones convexas (cuya unión forma las restricciones originales) y obtener la solución explícita de cada uno de ellos. A continuación, se intersectan las particiones y se analizan las posibles soluciones en cada región resultante.
En la sección 4.2 se ha obtenido la solución para el caso de restricciones formadas por la unión de dos poliedros. Se han formulado diferentes propiedades y proposiciones para simplificar la intersección de las dos particiones. Además, se han introducido procedimientos para la división de regiones en función de cuál de las dos posibles soluciones en cada una de ellas es la óptima. En primer lugar, se han propuesto métodos basados en programación cuadrática y/o lineal para los casos particulares que cumplen las propiedades vistas en el capítulo 3. En segundo lugar, se ha introducido un procedimiento general, basado en programación de suma de cuadrados, que permite la resolución de la factibilidad de sistemas definidos mediante ecuaciones polinómicas. Por último, se ha introducido un algoritmo que permite, utilizando resultados de fases anteriores, definir la unión de regiones cuya solución es la misma mediante un número mínimo de poliedros.
En la sección siguiente, se ha extendido el procedimiento presentado para el caso en que las restricciones no convexas estén formadas por la unión de un número mayor de poliedros. Se ha utilizado una algoritmo progresivo, que obtiene primero la solución considerando sólo dos regiones poliédricas de restricciones y que posteriormente va añadiendo más particiones del espacio de estados. Se ha mostrado como cada paso del algoritmo es análogo al problema resuelto en la sección anterior, aunque las proposiciones que permiten la simplificación del proceso de intersección de particiones son diferentes. Además, se ha generalizado el algoritmo para la eliminación de soluciones no óptimas para el caso de un mayor número de soluciones.

En la sección 4.4, se ha introducido una metodología diferente, basada en el cálculo de la envolvente convexa de las restricciones. Se ha demostrado que, dado un valor de las variables de estado $x$, el óptimo que se obtiene para un nuevo problema sujeto a unas restricciones definidas como la envolvente convexa de las restricciones originales tiene siempre un coste igual o menor que el óptimo del problema real. Usando esta propiedad, se ha obtenido la solución explícita del problema simplificado y se han clasificado las regiones en tres grupos diferentes: las que coinciden exactamente con regiones de la solución del problema real, las que están en el interior de alguna de dichas regiones y el resto. Se ha demostrado que sólo las regiones del último tipo pueden tener más de un óptimo, y por tanto puede ser necesario dividirlas. Se ha planteado un algoritmo para ello, basado también en programación de suma de cuadrados. La metodología, al igual que la otra alternativa planteada,

finaliza con un procedimiento de unión de las regiones con una misma expresión afín del óptimo.

Por último, se han comparado los dos algoritmos planteados en términos de coste computacional fuera de línea. Se ha demostrado que, aunque se trata de un resultado dependiente del problema, para la mayoría de los casos es más conveniente el algoritmo basado en la envolvente convexa.

Capítulo 5

# Algoritmo en línea

## 5.1. Introducción

En el capítulo 3 se ha estudiado la forma que tiene la solución explícita de un problema de control predictivo con restricciones formadas por la unión no convexa de varios poliedros, mostrándose que se trata de una solución afín a tramos, definida sobre una partición del espacio de estados con regiones descritas mediante desigualdades lineales y cuadráticas. Por otra parte, en el capítulo 4 se han propuesto diferentes métodos para calcular dicha solución explícita.

En el presente capítulo, se estudiará la forma ideal de implementar el algoritmo en línea que, para un determinado vector de estados $x$, calcula la acción de control óptima a partir de una partición calculada mediante alguno de los procedimientos expuestos en el capítulo 4. La propuesta de algoritmo en línea se basará en los métodos de árboles de búsqueda binarios utilizados para sistemas con restricciones lineales (descritos en el capítulo 2) por ser éstos los que ofrecen un menor coste computacional para este tipo de sistemas. Es interesante observar que, si ya en el caso de restricciones poliédricas convexas es importante diseñar algoritmos en línea adecuados para reducir el coste computacional, en el caso de tener restricciones formadas por unión no convexa de poliedros todavía lo es más, especialmente si se desea trabajar con períodos de muestreo bajos. Esto es así porque, al estar definidas las regiones por curvas cuadráticas además de desigualdades lineales, las comprobaciones a realizar requerirán un mayor número de operaciones.

Con respecto al algoritmo en línea, en el capítulo se tratan las siguientes cuestiones:

- En primer lugar, se describen los árboles binarios y un algoritmo para recorrerlos, analizando el coste computacional que ello supone en el caso de tener regiones lineales.

- Posteriormente se describe la manera de obtener, para una solución explícita dada, el árbol binario de profundidad mínima, así como el coste en línea del algoritmo.
- A continuación, se estudian los problemas derivados de utilizar dicho árbol de profundidad mínima: el coste computacional fuera de línea es elevado y no existen garantías de que el coste en línea sea mínimo, por el coste superior de evaluar una desigualdad cuadrática respecto a una lineal. Por tanto, se propone un nuevo algoritmo en línea que, con un coste fuera de línea moderado, ofrece mayores garantías de reducción del coste en línea.

Por último, debido a que el coste en línea de los algoritmos puede ser una restricción importante para la aplicación de las técnicas de control predictivo que se proponen en la tesis, en la última sección del capítulo se plantean algunas simplificaciones que llevan a soluciones subóptimas. Se distinguirán dichas simplificaciones en dos grupos: simplificación del problema de optimización y simplificación de la solución explícita.

## 5.2. Árboles de búsqueda binarios

Los árboles de búsqueda binarios, descritos en la sección 2.3, fueron introducidos en [TJB03b] para determinar en cuál de las regiones de una partición lineal del espacio de estados se encuentra un determinado vector de estados $x$. En el contexto del control predictivo con restricciones lineales, determinar dicha región supone conocer cuál es la acción de control a aplicar al sistema.

El árbol binario es una estructura de datos formada por varios nodos, cada uno de los cuales, a excepción de los nodos hoja, tiene asociada una determinada desigualdad $d(x)$ y dos nodos hijos. Cuando se recorre el árbol para un valor dado de $x$, la desigualdad asociada a cada nodo tomará un signo, que indica cuál de los dos nodos hijos debe tomarse. Procediendo de esta forma, se recorre el árbol binario hasta llegar a un determinado nodo hoja. Éstos, por su parte, tienen asociada una ley de control a aplicar. El algoritmo 5.1 resume el procedimiento para recorrer el árbol.

**Algoritmo 5.1** Recorrido del árbol binario

1. Asignar al nudo actual, $N_k$, el nudo raíz.
2. Mientras $N_k$ no sea un nudo hoja:
   *a*) Evaluar la desigualdad $d(x)$ correspondiente a $N_k$.
   *b*) Dependiendo del signo de $d(x)$ asignar a $N_k$ el hijo del nudo actual correspondiente.
3. Evaluar la ley de control $u(x)$ correspondiente a $N_k$.

En el caso de la partición lineal, las desigualdades que aparecen en cada uno de los nodos se corresponden con alguno de los hiperplanos que definen las regiones de la partición, de forma que cada uno de sus dos nodos hijos contiene la información de qué regiones se encuentran a cada uno de los dos lados de dicho hiperplano. Los nodos hoja corresponden a una única región, por lo que se conoce cuál es la ley de control afín que les corresponde.

A continuación, y con objeto de comparar posteriormente los algoritmos en línea que se plantean, se calcula el coste computacional del algoritmo anterior para el caso de una partición lineal. Para ello, se tiene en cuenta que, en cada nodo no-hoja, hay que comprobar el signo una desigualdad lineal $d(x) = L_j x - W_j$ lo que supone realizar $n$ productos, $n$ sumas y una comparación, es decir, $2n + 1$ operaciones elementales, siendo $n$ el número de variables de estado. En el nodo hoja, en el que se encuentra la ley de control a aplicar, hay que calcular $u(x) = G_i x + h_i$ lo que implica realizar, para cada elemento del vector $u$, $n$ productos y $n$ sumas. Si se tienen $m$ acciones de control, el número de operaciones elementales a realizar en el nodo hoja son $2nm$. Si se define la profundidad del árbol $D$ como la distancia entre los nodos hoja y el nodo raíz, éste es el número de nodos no-hoja que hay que atravesar en cada recorrido del árbol, por lo que el coste total del algoritmo de recorrido del árbol es:

$$C = (2n + 1)D + 2nm$$

La profundidad del árbol $D$, para un caso ideal, puede calcularse suponiendo que cada nodo tiene asociado un hiperplano que deja la mitad de regiones a un lado y la mitad al otro. En ese caso, teniendo $N$ regiones, $D$ puede calcularse como:

$$D = \lceil log_2(N) \rceil$$

No obstante, dicho caso ideal sólo será posible para particiones del espacio de estados muy específicas, por lo que en [TJB03b] se introduce un parámetro $\alpha \in [0{,}5, 1)$ para representar el equilibrado del árbol (siendo $\alpha = 0{,}5$ el caso ideal en que en cada nodo se dividen exactamente todas las regiones restantes). Con ello, la profundidad del árbol puede calcularse como:

$$D = \left\lceil -\frac{lnN}{ln\alpha} \right\rceil$$

Uno de los aspectos que puede hacer inconveniente el uso de árboles binarios frente a la optimización en línea es la mayor necesidad de memoria de los primeros. Los elementos que será necesario almacenar para poder ejecutar el algoritmo de recorrido del árbol binario son:

- Las $N$ diferentes leyes de control (almacenadas cada una de ellas mediante una matriz de tamaño $[m \times n]$ y un vector de tamaño $[m \times 1]$).
- Los $L$ hiperplanos diferentes que definen las regiones de la partición (cada uno de ellos almacenado mediante un vector de tamaño $[1 \times n]$ y un escalar)

- Para cada nodo no-hoja, un puntero al hiperplano asociado y dos punteros más: a cada uno de los dos nodos hijos si éstos son también nodos no-hoja, o a las dos leyes de control posibles, si el nodo el cuestión es el último no-hoja.

Por tanto, para conocer los requerimientos de memoria, será necesario conocer el número de nodos en el árbol. Es imposible conocer dicho número sin construir el árbol, aunque puede estimarse una cota superior suponiendo que el árbol es completo (es decir, que la profundidad es la misma para todos los nodos hoja). En este caso, el número de nodos aproximados es:

$$2^D = 2^{\lceil -\frac{lnN}{ln\alpha} \rceil} \approx N^{\frac{-ln2}{ln\alpha}}$$

En el presente capítulo, se pretende seguir la misma filosofía para determinar en qué región de la partición de regiones lineales y cuadráticas se encuentra una $x$ dada, obteniendo de esta forma la acción de control a aplicar para el problema de control predictivo con restricciones no convexas (3.3). La principal dificultad para ello estriba en que, si bien para los problemas con restricciones lineales las desigualdades $d(x)$ son lineales, para el caso de restricciones no convexas se trata de funciones cuadráticas (que además en general pueden ser indefinidas). Como se mostrará a continuación, este hecho aumenta considerablemente el coste computacional tanto del algoritmo 5.1 como de los procedimientos para la construcción del árbol.

## 5.3. Árboles binarios de los subproblemas

Como se vio en la sección 3.3, una primera opción para la obtención de la solución del problema de optimización (3.3), cuyas restricciones son la unión no convexa de $\gamma$ regiones poliédricas, es formular los $\gamma$ subproblemas de optimización (5.1), resolverlos en línea, y buscar aquél cuyo índice de coste $V_i^{OPT}(x)$ sea menor.

$$\mathcal{P}(x): \quad V^{OPT}(x) := \min \left\{ V_i^{OPT}(x) \right\}$$

donde:

$$V_i^{OPT}(x) := \min \tfrac{1}{2}\mathbf{u}^T H \mathbf{u} + \mathbf{u}^T F x, \tag{5.1}$$

sujeto a:
$\mathbf{u} \in T_i$

No obstante, si se desea evitar la resolución de los QPs en línea, una primera posibilidad sería obtener la solución explícita de cada subproblema fuera de línea y, una vez se esté trabajando en línea, seguir el procedimiento 5.2 para la obtención de la acción de control a aplicar.

Para el algoritmo en línea de búsqueda de las regiones en las que se encuentra el vector de estados $x$, dado que cada una de las particiones es lineal, como se discutió en el capítulo 2 la opción menos costosa es la construcción de un

**Algoritmo 5.2** Algoritmo en línea con $\gamma$ árboles binarios

1. Para cada $T_i$, buscar en qué región $R_{ij}$ de la partición se encuentra $x$ y obtener el valor del índice de coste en el óptimo $V_i^{OPT}(x)$.
2. Buscar el índice $V_i^{OPT}(x)$ mínimo.
3. Aplicar la ley de control $u(x)$ correspondiente a la región cuyo índice de coste es mínimo.

árbol binario y su recorrido mediante el algoritmo 5.1. En cuanto al cálculo del índice de coste para cada subproblema, éste requerirá realizar la siguiente serie de operaciones:

- $n$ productos de vectores $[1 \times n]$ por $[n \times 1]$, uno por cada fila de la matriz $Mx$.
- 1 producto de vector $[1 \times n]$ por $[n \times 1]$, correspondiente al producto de los vectores $x^T$ y $Mx$.
- 1 producto de vector $[1 \times n]$ por $[n \times 1]$, del producto $Nx$.
- 2 sumas, $x^T Mx + Nx$ y el resultado anterior más $C$.

Teniendo en cuenta que el coste de cada producto de vectores $[1 \times n]$ por $[n \times 1]$ es de $2n - 1$ operaciones elementales ($n$ productos y $n - 1$ sumas) el coste del cálculo de un índice de coste es de $n(2n - 1) + 2(2n - 1) + 2 = n(2n + 3)$. Para simplificar las operaciones, definimos el coste de evaluar el signo de una desigualdad lineal $Lx + W < 0$ como $c_0 = 2n + 1$. De esta forma, el coste de cada una de las operaciones del algoritmo 5.2 es:

- Recorrer cada árbol binario: $c_0 D_i$
- Calcular el valor de cada índice de coste: $n(2n + 3) = n(c_0 + 2)$.
- Buscar el mínimo de los $\gamma$ índices de coste: $\gamma - 1$
- Aplicar la ley de control correspondiente: $2nm$.

De esta forma, el coste total del algoritmo en línea 5.2 puede expresarse como:

$$C_1 = c_0 \left( \sum_{i=1}^{\gamma} D_i + \gamma(n + 1) \right) + 2nm - 1 \tag{5.2}$$

En cuanto a los requerimientos de memoria del algoritmo, éstos son:

- $\sum_{i=1}^{\gamma} N_i$ leyes de control (requiriendo cada una de ellas una matriz $F_i$ de tamaño $[m \times n]$ y un vector $G_i$ de tamaño $[m \times 1]$).

- $\sum_{i=1}^{\gamma} N_i$ índices de coste (requiriendo cada una de ellos una matriz $M_i$ de tamaño $[n \times n]$, un vector $N_i$ de tamaño $[n \times 1]$ y una constante $C_i$).
- $\sum_{i=1}^{\gamma} L_i$ hiperplanos (requiriendo cada una de ellos un vector $L_i$ de tamaño $[n \times 1]$ y una constante $W_i$)
- $3\sum_{i=1}^{\gamma} 2^{D_i-1}$ punteros, 3 en cada nodo no-hoja de cada árbol.

## 5.4. Árbol binario de profundidad mínima

Como se vio en la sección 2.3, cuando todas las desigualdades que definen una partición del espacio de estados son lineales, el criterio fundamental para el diseño del árbol binario para el algoritmo en línea es aquel que minimiza la profundidad $D$, puesto que este parámetro afecta de manera lineal al coste del algoritmo.
En esta sección, a partir de la idea de [TJB03b], se va a plantear la obtención del árbol binario de profundidad mínima para el caso de la partición del espacio de estados que se obtiene como solución del problema (3.3).

### 5.4.1. Obtención del árbol

Dado el problema de control predictivo con restricciones no convexas, se ha visto que la solución explícita consiste en un controlador afín a tramos (3.9) definido sobre regiones caracterizadas por desigualdades lineales y cuadráticas de la forma (3.10). Consideramos el conjunto de todas las desigualdades que definen las diferentes regiones $X_i$:

$$d_j(x) = \begin{cases} L_j x - W_j & \text{para } j = 1, \ldots, L \\ x^T M_j x + N_j x + C_j & \text{para } j = L+1, \ldots, L+q \end{cases} \tag{5.3}$$

y definimos:

- La representación de índices de un conjunto semialgebraico, $\mathcal{J}$, como una combinación de índices y signos de $d_j$. Por ejemplo, $\mathcal{J} = \{1^+, 2^-, 5^+\}$ representaría al conjunto semialgebraico definido por $d_1 \geq 0$, $d_2 \leq 0$ y $d_5 \geq 0$.
- El conjunto de regiones correspondientes a $\mathcal{J}$, $\mathcal{I}$. Se trata de todas las regiones $X_i$ que satisfacen todas las restricciones definidas en $\mathcal{J}$.
- El conjunto de funciones afines correspondientes a un conjunto $\mathcal{I}$: $\mathcal{F}(\mathcal{I}) = \{k \,|\mathcal{K}_k$ corresponde a $X_i, i \in \mathcal{I}\}$.

La idea fundamental para la construcción de un árbol de profundidad mínima es la misma que para el caso de las regiones lineales: se pretende que la desigualdad que se asigna a cada nodo $N_k$ divida la parte del árbol que

queda por debajo en dos partes lo más iguales posibles. Es decir, idealmente se pretende elegir una desigualdad $j_k$ tal que el número de funciones afines válidas para regiones que satisfacen todas las restricciones de nodos anteriores, $|\mathcal{F}(\mathcal{I}(\mathcal{J}_k))|$, quede dividido en dos subconjuntos con el mismo número de regiones: $|\mathcal{F}(\mathcal{I}(\mathcal{J}_k \cup j_k^+))| = |\mathcal{F}(\mathcal{I}(\mathcal{J}_k \cup j_k^-))| = \frac{1}{2}|\mathcal{F}(\mathcal{I}(\mathcal{J}_k))|$. Como en un caso general no es posible dividir el conjunto $|\mathcal{F}(\mathcal{I}(\mathcal{J}_k))|$ en dos subconjuntos con exactamente el mismo número de elementos, se intenta que el mayor número de elementos de un subconjunto sea lo menor posible: $j_k = \arg\min_j \max(|\mathcal{F}(\mathcal{I}(\mathcal{J}_k \cup j_k^+))|, |\mathcal{F}(\mathcal{I}(\mathcal{J}_k \cup j_k^-))|)$.
Nótese en este punto que $\mathcal{I}(\mathcal{J}_k \cup j_k^\pm) \subseteq (\mathcal{I}(\mathcal{J}_k) \cap \mathcal{I}(j\pm))$, es decir, las regiones $X_i$ que satisfacen simultáneamente todas las restricciones en $\mathcal{J}_k$ además de $j_k^\pm$, son un subconjunto de las regiones que satisfacen por un lado $\mathcal{J}_k$ y por el otro $j_k^\pm$. Por tanto, una primera aproximación de $\mathcal{I}(\mathcal{J}_k \cup j_k^\pm)$, sencilla de calcular, puede ser $(\mathcal{I}(\mathcal{J}_k) \cap \mathcal{I}(j^\pm))$.

Aplicando el lema 1 en [TJB03b], cuya demostración es válida aunque las desigualdades que definen las regiones no sean lineales, se deduce que las regiones $X_i$ que constituyen la diferencia entre los conjuntos $\mathcal{I}(\mathcal{J}_k \cup j_k^\pm)$ e $(\mathcal{I}(\mathcal{J}_k) \cap \mathcal{I}(j^\pm))$ siempre están divididas por la restricción $j_k$, es decir pertenecen a $(\mathcal{I}(\mathcal{J}_k) \cap \mathcal{I}(j^+) \cap \mathcal{I}(j^-))$.

El algoritmo 5.3 detalla el procedimiento descrito para la construcción del árbol binario. Dado que el principio para la construcción del árbol binario, conseguir que éste sea de profundidad mínima, coincide con el planteado para el caso de particiones con restricciones lineales, el algoritmo es exactamente igual al que se expuso en la sección 2.3. No obstante, existe una diferencia fundamental en el modo de resolver las operaciones necesarias para la construcción del árbol, esto es, el cálculo de $\mathcal{I}(j^\pm)$ e $(\mathcal{I}(\mathcal{J}_k) \cap \mathcal{I}(j^\pm))$.

Para el caso de restricciones lineales, esos conjuntos de índices de regiones se obtenían mediante la resolución de LPs. En el caso que nos ocupa, dada una desigualdad (lineal o cuadrática) $d_j(x)$ y una región $X_i$ definida como:

$$X_i = \left\{ x \in \mathbb{R}^n \;\middle|\; \begin{array}{c} L_i x \leq W_i \\ x^T M_{ik} x + N_{ik} x + C_{ik} \leq 0 \quad \text{para } k = 1, \ldots, q_k \end{array} \right\}$$

calcular los conjuntos $\mathcal{I}(j^\pm)$ equivale a comprobar la factibilidad de los sistemas de desigualdades:

$$\left\{ \begin{array}{c} L_i x \leq W_i \\ x^T M_{ik} x + N_{ik} x + C_{ik} \leq 0 \quad k = 1, \ldots, q_k \\ d_j(x) \geq 0 \end{array} \right.$$

y

$$\left\{ \begin{array}{c} L_i x \leq W_i \\ x^T M_{ik} x + N_{ik} x + C_{ik} \leq 0 \quad k = 1, \ldots, q_k \\ d_j(x) \leq 0 \end{array} \right.$$

Es decir, es necesario resolver, para cada pareja de desigualdad y región, dos problemas SOS.

**Algoritmo 5.3** Construcción del árbol binario

1. Calcular los conjuntos de índices $\mathcal{I}(j^+)$ e $\mathcal{I}(j^-)$ para cada $j = \{1, \ldots, L\}$.
2. Inicializar el nodo raíz del árbol como $N_1 \leftarrow (\{1, \ldots, n_r\}, \emptyset)$
3. Inicializar el conjunto de nodos inexplorados como $\mathcal{U} \leftarrow \{N_1\}$.
4. Seleccionar cualquier nodo inexplorado $N_k \in \mathcal{U}$ y hacer $\mathcal{U} \leftarrow \mathcal{U} \setminus N_k$.
5. Calcular las aproximaciones $\mathcal{I}(\mathcal{J}_k) \cap \mathcal{I}(j^{\pm})$ para cada j, y ordenar las desigualdades por la cantidad $max(|\mathcal{F}(\mathcal{I}(\mathcal{J}_k) \cap \mathcal{I}(j^+))|, |\mathcal{F}(\mathcal{I}(\mathcal{J}_k) \cap \mathcal{I}(j^-))|)$.
6. Calcular exactamente $\mathcal{I}_k^{\pm} = \mathcal{I}(\mathcal{J}_k \cap j^{\pm})$ para cada uno de los $n_j$ primeros elementos en la lista ordenada del paso anterior. Esto se hace resolviendo los problemas SOS (5.4) y (5.5) para cada $i \in \mathcal{I}(\mathcal{J}_k) \cap \mathcal{I}(j^+) \cap \mathcal{I}(j^-)$. Seleccionar entre ellos $j_k = argmin_j max(|\mathcal{F}(\mathcal{I}_k^+)|, |\mathcal{F}(\mathcal{I}_k^-)|)$.
7. Completar el nodo como $N_k \leftarrow j_k$, y crear dos nodos hijos, $N^{\pm} \leftarrow (\mathcal{I}_k^{\pm}, \mathcal{J}_k \cup j^{\pm})$.
8. Si $|\mathcal{F}(\mathcal{I}^{\pm})| > 1$, añadir $N^{\pm}$ a $\mathcal{U}$. Si no, $N^{\pm}$ es un nodo hoja, hacer $N^{\pm} \leftarrow \mathcal{F}(\mathcal{I}^{\pm})$.
9. Si $\mathcal{U} \neq \emptyset$, volver al paso 4. Si no, finalizar.

En cuanto a la obtención del conjunto $\mathcal{I}(\mathcal{J}_k \cup j_k^+)$, como ocurría con las restricciones lineales es conveniente hacer uso del lema 1 en [TJB03b]. Por la aplicación de éste, sabemos que para cualquier $i \in (\mathcal{I}(\mathcal{J}_k) \cap \mathcal{I}(j_k^+))$ que cumpla $i \notin (\mathcal{I}(\mathcal{J}_k) \cap \mathcal{I}(j_k^+) \cap \mathcal{I}(j_k^+))$, se tiene $i \in \mathcal{I}(\mathcal{J}_k \cup j_k^+)$. Por tanto, únicamente es necesario resolver un problema de optimización para saber si $i \in \mathcal{I}(\mathcal{J}_k \cup j_k^+)$ para aquellas regiones $X_i$ que se encuentren simultáneamente en $\mathcal{I}(\mathcal{J}_k)$, $\mathcal{I}(j_k^+)$ e $\mathcal{I}(j_k^-)$. Dicho problema puede formularse como la comprobación de factibilidad del sistema de desigualdades (5.4), que de nuevo puede realizarse mediante la resolución de un problema SOS para cada región $X_i$:

$$\left\{ \begin{array}{cc} L_i x \leq W_i & \\ x^T M_{ik} x + N_{ik} x + C_{ik} \leq 0 & k = 1, \ldots, q_k \\ d_j(x) \geq 0 & j \in \mathcal{J}_k^+ \\ d_j(x) \leq 0 & j \in \mathcal{J}_k^- \\ d_{j_k}(x) \geq 0 & \end{array} \right. \tag{5.4}$$

donde $\mathcal{J}_k^+$ y $\mathcal{J}_k^-$ representan, respectivamente, los índices con signo positivo y negativo de $\mathcal{J}_k$.

La obtención del conjunto $\mathcal{I}(\mathcal{J}_k \cup j_k^-)$, también necesario en el algoritmo, se realiza de manera análoga comprobando para cada $X_i$ la factibilidad de (5.5).

$$\left\{\begin{array}{cc} L_i x \leq W_i & \\ x^T M_{ik} x + N_{ik} x + C_{ik} \leq 0 & k = 1, \ldots, q_k \\ d_j(x) \geq 0 & j \in \mathcal{J}_k^+ \\ d_j(x) \leq 0 & j \in \mathcal{J}_k^- \\ d_{j_k}(x) \leq 0 & \end{array}\right. \tag{5.5}$$

Como ya ocurre en el algoritmo para regiones lineales, pero con mayor importancia en este caso por el elevado coste computacional de la resolución de un problema SOS en comparación con un LP, se introduce en el paso 6 del algoritmo el parámetro de diseño $n_j$. Este parámetro indica para cuántas regiones de de la lista ordenada de $\mathcal{I}(\mathcal{J}_k) \cap \mathcal{I}(j_k^{\pm})$ se comprueba de manera exacta si se encuentran en $\mathcal{I}(\mathcal{J}_k \cup j_k^{\pm})$. Por tanto, se trata de un parámetro que muestra el compromiso entre el coste del algoritmo de construcción del árbol binario (fuera de línea) y la profundidad de dicho árbol.

Por último, es necesario establecer un criterio para la selección de una desigualdad $j_k$ en el paso 6 cuando la solución que minimiza el criterio descrito no es única. Para el caso de restricciones lineales, los autores de [TJB03b] proponen dos criterios diferentes, uno de ellos para reducir el número de regiones en cada nodo y otro para reducir el número de nodos en todo el árbol, con el objetivo de reducir la memoria necesaria para el algoritmo. No obstante, para el caso planteado en el que existen restricciones lineales y cuadráticas, se elegirá siempre que sea posible una *desigualdad lineal antes que una cuadrática*. Esto es debido a que, como se justificará a continuación, el coste en línea puede ser menor, puesto que éste no depende únicamente de la profundidad del árbol. En el caso de existir únicamente desigualdades de un tipo (lineal o cuadrática), puede aplicarse cualquiera de los otros dos criterios.

### 5.4.2. Coste computacional en línea

El número de operaciones elementales que es necesario realizar para el cálculo de la acción de control es el correspondiente a la comprobación de una desigualdad en cada nodo no-hoja del árbol y el cálculo de la acción de control óptima al llegar al nodo hoja adecuado. No obstante, el coste en cada nodo no-hoja será diferente si la desigualdad a comprobar es cuadrática o lineal:

- Desigualdades lineales: Como se ha visto anteriormente, el coste es de $c_0 = 2n + 1$ operaciones elementales.
- Desigualdades cuadráticas: El coste es el de evaluación de una función cuadrática $n(c_0 + 2)$ más una comparación, en total $(n + 1)c_0$.

Por tanto, para la obtención del coste en línea del algoritmo es necesario definir dos profundidades distintas, la del número de desigualdades lineales, $D_l$, y el de

cuadráticas en el peor caso $D_q$. El coste computacional total queda por tanto:

$$C_2 = c_0 \left((n+1)D_q + D_l\right) + 2nm \tag{5.6}$$

A la vista de la expresión del coste, se justifica el criterio tomado en el diseño del algoritmo consistente en elegir preferentemente restricciones lineales en lugar de cuadráticas, por el coste mucho más elevado de comprobación de estas últimas.

Es interesante observar también que, a diferencia del problema con restricciones lineales, obtener una profundidad mínima no garantiza un coste en línea del algoritmo mínimo. No obstante, sí es el algoritmo que proporciona un mínimo de nodos en el árbol.

En cuanto a los requerimientos de memoria del algoritmo, éstos son:

- Un máximo de $\sum_{i=1}^{\gamma} N_i$ leyes de control. En el caso de que, al obtener la partición del estado final, la ley de control de alguna de las regiones de una de las particiones originales resulte siempre en un índice de coste mayor que el correspondiente a alguna de las leyes de control de las regiones del resto de particiones, no es necesario almacenar dicha ley.
- $q$ curvas cuadráticas. En un caso general, $q \leq N_r \binom{\gamma}{2}$. Aunque en el peor caso esto implica el almacenamiento de más curvas cuadráticas que si se guardaran los índices de coste, habitualmente muchas de las curvas serán no factibles en las regiones, y por tanto $q$ será mucho menor.
- $L \leq \sum_{i=1}^{\gamma} L_i$ hiperplanos. Debido a la unión de regiones que se produce al obtener la solución final, alguno de los hiperplanos puede ser redundante en todas las regiones, con lo que no sería necesario almacenarlo.
- $3 \cdot 2^{D_l + D_q - 1}$ punteros, 3 en cada nodo no-hoja de cada árbol.

## 5.5. Árboles binarios de profundidad no mínima

En la sección anterior se ha visto como, a diferencia del problema de control predictivo con restricciones politópicas, cuando las restricciones no son convexas una profundidad mínima del árbol binario no garantiza un coste en línea mínimo. Teniendo en cuenta además, que el algoritmo fuera de línea para la obtención del árbol requiere la resolución de un número elevado de problemas de optimización, en esta sección se va a plantear una estrategia diferente.

La idea consiste en realizar el árbol binario, en lugar de sobre la partición final de regiones solución al problema planteado, sobre una partición lineal cuyas regiones no definen todavía la expresión afín del controlador, sino que tienen asociadas varias soluciones posibles. El último paso del algoritmo consistirá en obtener cual de esas soluciones es la óptima.

La motivación fundamental de un algoritmo de este tipo es que, como se comprobará a continuación, el coste de obtención del árbol binario es muy reducido en comparación con el coste del árbol de profundidad mínima.

### 5.5.1. Algoritmo fuera de línea

Para la obtención del algoritmo se tendrá en cuenta que, tanto si para el cálculo de la solución explícita se ha empleado la metodología de intersección, división y unión de la sección 4.3 como si se ha empleado la metodología de la envolvente convexa de la sección 4.4, se han obtenido:

- Una partición lineal del espacio de estados.
- Para cada región lineal, las diferentes soluciones afines que son óptimas en algún punto de la región.

Con esta información disponible, se plantea al algoritmo fuera de línea 5.4. El único paso de dicho algoritmo que requiere operaciones costosas es el 1, construcción del árbol binario de las regiones lineales. Para ello, se sigue el algoritmo 2.5 que, como se vio, requiere la resolución de un determinado número de problemas LP. No obstante, es conveniente tener en cuenta que para el cálculo de muchos de los conjuntos $\mathcal{I}(j^{\pm})$, los problemas LP necesarios ya habrán sido resueltos para el cálculo de las marcas de las regiones necesarias para la aplicación de los algoritmos de minimización del número de regiones 4.8.

**Algoritmo 5.4** Árbol de profundidad no mínima. Algoritmo fuera de línea.

1. Construcción del árbol binario de las regiones lineales aplicando el algoritmo 2.5.
2. Para cada región lineal, comprobar el número de soluciones afines posibles,$n_s$:
   - Si $n_s = 1$, almacenar únicamente esa solución factible.
   - Si $n_s = 2$, almacenar las dos soluciones afines y la curva cuadrática resta de las dos funciones de coste $q(x) = J_1(x) - J_2(x)$.
   - Si $n_s > 2$, almacenar cada solución afín y cada índice de coste.

Es importante señalar que el algoritmo 5.4 no requiere la resolución de ningún problema SOS (adicional a los ya resueltos para la obtención de la solución explícita). Por tanto, el coste computacional será considerablemente menor al que tiene el algoritmo para la obtención del árbol de profundidad mínima.

### 5.5.2. Coste computacional en línea

El procedimiento en línea a seguir consiste en recorrer el árbol binario de las regiones lineales y, en función del número de soluciones existentes en la

región lineal obtener la solución afín óptima. El algoritmo 5.5 resume dicho procedimiento. Es interesante observar que, para $n_s > 2$, sería posible obtener

**Algoritmo 5.5** Árbol de profundidad no mínima. Algoritmo en línea.

1. Recorrido del árbol binario de las regiones lineales aplicando el algoritmo 5.1.
2. Para cada región lineal, comprobar el número de soluciones afines posibles,$n_s$:
   - Si $n_s = 1$, aplicar el controlador afín correspondiente.
   - Si $n_s = 2$, comprobar a qué lado de la curva $q(x)$ se está y aplicar el controlador afín correspondiente.
   - Si $n_s > 2$, calcular todos los índices de coste, buscar el mínimo y aplicar el controlador afín correspondiente.

curvas $q_{ij} = J_i - J_j$ de manera análoga a la propuesta para el caso de $n_s = 2$. Hacerlo de esta forma permitiría, con un procedimiento análogo al descrito para la obtención del árbol binario de profundidad mínima, el cálculo de un segundo árbol binario dentro de la región lineal que indicara el modo más conveniente de comprobar las curvas cuadráticas. No obstante, como se ha visto esto supondría la resolución de problemas SOS costosos computacionalmente y, siempre que el número $\gamma$ de poliedros que definen las restricciones no sea muy alto, no supondrá una disminución significativa del coste computacional en línea.

Si no se realiza ese segundo árbol binario descrito, es más conveniente almacenar los $n_s$ índices de coste $J_i(x)$ que las $\binom{n_s}{2} \geq n_s$ curvas $q_{ij}(x)$. Aunque comprobar las segundas requiere el cálculo de una forma cuadrática menos, requiere un espacio de almacenamiento considerablemente mayor.

Para analizar el coste del algoritmo, se enumeran todas las operaciones que hay que realizar en el peor caso (recorrer el máximo número de nodos en el árbol, $D$, y llegar a una región con el máximo de soluciones posibles, $n_s^{max}$):

- Recorrido del árbol binario de regiones lineales: $c_0 D$.
- Cálculo de los $n_s^{max}$ índices de coste: $n_s^{max} n(c_0 + 2)$.
- Comparaciones para búsqueda del mínimo: $n_s^{max} - 1$.
- Aplicación del controlador afín óptimo: $2nm$.

Con todo esto, el coste en línea del algoritmo es:

$$C_3 = c_0 \left(D + n_s^{max}(n+1)\right) + 2nm - 1 \tag{5.7}$$

En cuanto a los requerimientos de memoria del algoritmo, éstos son:

- Un máximo de $\sum_{i=1}^{\gamma} N_i$ leyes de control.
- Un máximo de $\sum_{i=1}^{\gamma} N_i$ índices de coste.
- $L \leq \sum_{i=1}^{\gamma} L_i$ hiperplanos.
- $3 * 2^{D-1}$ punteros, 3 en cada nodo no-hoja de cada árbol.

## 5.6. Comparación de los algoritmos

En está sección se comparar los tres algoritmos propuestos en sus tres facetas fundamentales: coste computacional en línea, necesidad de memoria para la implementación en línea y coste computacional fuera de línea.

### 5.6.1. Coste computacional en línea

Inicialmente, se comparará el algoritmo de árboles binarios independientes, cuyo coste $C_1$ viene dado por (5.2), con el de profundidad no mínima, de coste $C_3$ dado por (5.7). Restando ambas expresiones se obtiene:

$$C_1 - C_3 = c_0 \left( \sum_{i=1}^{\gamma} D_i - D + (\gamma - n_s^{max})(n+1) \right)$$

De la definición de $n_s^{max}$, que expresa el número máximo de soluciones en una misma región lineal, se deduce que éste siempre será como máximo $\gamma$. Por tanto, se cumple $(\gamma - n_s^{max} \geq 0)$.

Por otra parte, en el peor caso, el número máximo de regiones lineales que se tendrá después de intersectar todas las particiones solución de los problemas convexos se obtiene si todas combinaciones posibles de regiones son factibles, $\prod^{\gamma} N_i$. Habitualmente, el número de regiones lineales será mucho menor, pudiendo expresarse como $N_t = \eta \prod^{\gamma} N_i$, con $\eta \leq 1$. Substituyendo en la expresión de la profundidad del árbol binario, se tiene:

$$D = \left\lceil -\frac{lnNt}{ln\alpha} \right\rceil = \left\lceil \frac{ln\eta + \sum_{i=1}^{\gamma} lnN_i}{-ln\alpha} \right\rceil$$

Teniendo en cuenta las propiedades de la función techo, $\lceil x \rceil + \lceil y \rceil - 1 \leq \lceil x+y \rceil \leq \lceil x \rceil + \lceil y \rceil$, se puede escribir:

$$\left\lceil \frac{ln\eta}{-ln\alpha} \right\rceil + \sum_{i=1}^{\gamma} \left\lceil \frac{lnN_i}{-ln\alpha} \right\rceil - \gamma \leq D \leq \left\lceil \frac{ln\eta}{-ln\alpha} \right\rceil + \sum_{i=1}^{\gamma} \left\lceil \frac{lnN_i}{-ln\alpha} \right\rceil$$

Considerando que $0{,}5 \leq \alpha < 1$ es un parámetro desconocido porque depende de cada partición en concreto, habitualmente se suele tomar una estimación

conservativa de $\alpha = 2/3$, por lo que podemos considerarlo constante. De esta forma, tenemos:

$$\left\lceil \frac{lnN_i}{-ln\alpha} \right\rceil = D_i$$

Substituyendo y operando a partir de la expresión anterior:

$$\left\lceil \frac{ln\eta}{-ln\alpha} \right\rceil + \sum_{i=1}^{\gamma} D_i - \gamma \leq D \leq \left\lceil \frac{ln\eta}{-ln\alpha} \right\rceil + \sum_{i=1}^{\gamma} D_i$$

$$-\left\lceil \frac{ln\eta}{-ln\alpha} \right\rceil \leq \sum_{i=1}^{\gamma} D_i - D \leq -\left\lceil \frac{ln\eta}{-ln\alpha} \right\rceil + \gamma$$

Considerando que $\lceil -x \rceil = -\lfloor x \rfloor$:

$$\left\lfloor \frac{ln\eta}{ln\alpha} \right\rfloor \leq \sum_{i=1}^{\gamma} D_i - D \leq \left\lfloor \frac{ln\eta}{ln\alpha} \right\rfloor + \gamma$$

Dado que $\eta \leq 1$ y $\alpha \leq 1$, $\left\lfloor \frac{ln\eta}{ln\alpha} \right\rfloor \geq 0$, por lo que se cumple $\sum_{i=1}^{\gamma} D_i - D \geq 0$.

De esta forma, queda demostrado que $C_1 - C_3 \geq c_0 \left\lfloor \frac{ln\eta}{ln\alpha} \right\rfloor \geq 0$, es decir, el algoritmo consistente en recorrer el árbol binario de las regiones lineales y buscar a continuación la función afín que produce un mínimo índice de coste tiene un coste computacional menor que el algoritmo consistente en recorrer todos los árboles binarios por separado y comparar los índices de coste de las diferentes soluciones.

Dado que el algoritmo del árbol binario de las regiones lineales ofrece un menor coste computacional, es el que se comparará con el del árbol binario de profundidad mínima. Restando ambos índices de coste, se tiene:

$$C_3 - C_2 = c_0 \left(D - D_l + (n_s^{max} - D_q)(n+1)\right) - 1$$

Se puede reescribir dicha diferencia como:

$$C_3 - C_2 = c_0 \left((D + n_s^{max} - D_q - D_l) + n(n_s^{max} - D_q)\right) - 1$$

Teniendo en cuenta que el árbol obtenido con el segundo algoritmo es de profundidad mínima, podemos asegurar que se cumple:

$$D_q + D_l \leq D + n_s^{max} - 1$$

Si la desigualdad anterior no se cumpliera, el árbol de profundidad mínima sería el que comprueba primero $D$ desigualdades lineales y, dentro de la región lineal, $n_s^{max} - 1$ curvas, y por tanto se tendría $D_q = n_s^{max} - 1$ y $D_l = D$.

Por tanto, podemos afirmar que el árbol de profundidad mínima tendrá un coste menor que el árbol obtenido con el algoritmo 5.4 siempre que se cumpla:

$$n_s^{max} \geq D_q$$

Aunque la condición anterior puede cumplirse en muchas ocasiones, no existen garantías de que el algoritmo de profundidad mínima tenga un coste computacional menor, y de hecho para algunos problemas éste será mayor que para el algoritmo de árbol binario de las regiones lineales.

### 5.6.2. Requerimientos de memoria

Otra faceta de importancia para comparar los diferentes algoritmos es el requerimiento de memoria para la implementación en línea.

En estos términos, el algoritmo 5.2 es el más desfavorable, puesto que requiere almacenar todas las leyes de control, hiperplanos e índices de coste de las regiones de cada uno de los subproblemas.

Los algoritmos 5.3 y 5.4 requieren el almacenamiento de las mismas leyes de control e hiperplanos. Puede ser que éstos sean todos los de los subproblemas, al igual que para el algoritmo de los árboles binarios por separado, pero también puede ocurrir que algunas de las leyes de control no sean óptimas en ninguna región y que algunos de los hiperplanos sean redundantes para todas las regiones lineales tras la unión de regiones, por lo que el requerimiento de almacenamiento sería menor.

En cuanto al número de curvas cuadráticas, el algoritmo 5.3 requiere almacenar:

$$\sum_{i=1}^{N_r} \binom{n_{si}}{2}$$

siendo $n_{si}$ el número de soluciones posibles para cada región lineal.

Para este algoritmo, si el problema en cuestión tiene muchas de las regiones lineales con una o dos soluciones posibles, será necesario almacenar muy pocas curvas cuadráticas. Por el contrario, si el problema tiene muchas regiones lineales con varias soluciones posibles, el número de curvas a almacenar puede ser incluso mayor que para el algoritmo de todos los árboles binarios por separado.

Para el algoritmo 5.4, se requieren $\sum_{i=1}^{N_r} n_{si}$ curvas cuadráticas. En este caso sí que se puede asegurar que el número de curvas cuadráticas a almacenar es menor que para el algoritmo de los árboles binarios por separado (puesto que como máximo existen $\sum^{\gamma} N_i$ índices de coste).

Por último, es necesario considerar el número de nodos de los diferentes árboles binarios, puesto que, como se ha visto, por cada uno de ellos es necesario almacenar tres punteros. Para el algoritmo de los árboles binarios por separado se tienen $\sum^{\gamma} 2^{D_i}$ nodos. Para el algoritmo del árbol binario de las regiones lineales $2^D$ y para el algoritmo de profundidad mínima $2^{(D_q+D_l)}$. Por tanto, este es el único término que puede ser más desfavorable para el algoritmo 5.4. No obstante, para la mayoría de implementaciones la memoria necesaria para almacenar un puntero es menor que la usada para almacenar un número real.

### 5.6.3. Coste fuera de línea

Como último criterio para comparar los diferentes algoritmos, se tienen en cuenta el número y tipo de problemas de optimización que es necesario resolver fuera de línea.

Bajo este criterio, el algoritmo de los árboles binarios por separado es claramente el más favorable, puesto que sólo requiere la resolución de LPs para la obtención de dichos árboles por separado, ahorrándose todos los problemas de optimización necesarios para la obtención de la solución explícita descritos en el capítulo 4.

En cuanto a los otros dos algoritmos, ambos necesitan obtener la solución explícita, con todos los problemas de optimización que ello requiere resolver. Aparte de éstos, el algoritmo del árbol binario de las regiones lineales requiere únicamente la resolución de los LPs para la obtención de dicho árbol, mientras que el de profundidad mínima requiere la resolución de problemas SOS, como se vio en la sección correspondiente. Por tanto, éste algoritmo es el más costoso fuera de línea.

### 5.6.4. Conclusiones

A la vista de los diferentes aspectos considerados, en un caso general en el que no se disponga información adicional del problema, el algoritmo más adecuado es el 5.4 puesto que es el que garantiza un coste computacional en línea y un requerimiento de memoria menor que el de los árboles binarios por separado, con un coste del algoritmo fuera de línea moderado.

El algoritmo 5.3 será el más adecuado cuando se sepa que los conjuntos de restricciones poliédricos tienen una disposición tal que en la mayoría de las regiones lineales habrá pocas soluciones posibles. En ese caso, el coste en línea y los requerimientos de memoria pueden ser menores que para el algoritmo 5.4 y además el coste fuera de línea no será demasiado elevado (puesto que, al haber pocas soluciones para cada región, el número de problemas SOS a resolver no será muy alto).

**Ejemplo 5.1.** *Se desea desarrollar un algoritmo en línea para la solución explícita obtenida en el ejemplo 4.10. Para ello, se plantean dos alternativas, el algoritmo 5.2, que obtiene los 4 árboles binarios correspondientes a cada uno de los problemas $\mathcal{P}_{T_i}$, y el árbol binario de profundidad no mínima, algoritmo 5.5.*

*En el primer caso, se tienen 4 soluciones explícitas de 9,21, 11 y 20 regiones. Se obtiene el árbol binario correspondiente a cada una de dichas soluciones con un total de, respectivamente, 35, 85, 45 y 95 nodos totales (18, 43, 23 y 48 de ellos, nodos hoja). Lo árboles quedan definidos mediante, respectivamente, 11, 32, 16 y 33 hiperplanos y tienen unas profundidades de $D_1 = 6$, $D_2 = 7$, $D_3 = 5$ y $D_4 = 7$. Aplicando* (5.2)*, se obtiene un coste computacional de $C_1 = 192$ operaciones elementales. En cuanto al uso de memoria, se requieren 1008 números reales y 516 punteros.*

*En cuanto al algoritmo de profundidad no mínima, como se vio en el ejemplo 4.10 se tiene una partición lineal con un total de 53 regiones. Al construirse el árbol binario, se obtienen 191 nodos (96 de ellos, nodos hoja) definidos mediante 63 hiperplanos y con una profundidad de $D = 9$. Aplicando* (5.7)*, y teniendo*

*en cuenta que $n_s^{max} = 2$, el coste computaciones en términos de operaciones elementales es de $C_3 = 82$. En cuanto a los requeremientos de memoria, es necesario almacenar 38 leyes de control, 12 funciones de coste y 63 hiperplanos, sumando un total de 489 números reales y 381 punteros.*

*Por tanto, el algoritmo 5.5 supone, respecto al algoritmo básico 5.2, una reducción del 57.2 % en el coste computacional. En cuanto al uso de memoria, suponiendo una implementación en un microcontrolador de 16 bits en el que los números reales necesitan el doble de memoria que los punteros, se consigue una reducción del 46.3 %.* ■

## 5.7. Soluciones subóptimas

Como se ha visto en las secciones anteriores, el coste del algoritmo en línea para la obtención de la acción de control óptima a aplicar es considerablemente más elevado que el derivado de obtener la acción de control óptima para un problema de control predictivo con restricciones convexas. Este hecho, en el contexto de procesos rápidos en los que es necesario un período de muestreo relativamente bajo, puede ser problemático.

Por otro lado, los algoritmos fuera de línea para la obtención de solución explícita, vistos en el capítulo 4, y para la obtención de árboles binarios, también puede requerir la resolución de muchos problemas de optimización SOS lo que puede ser, en determinados casos, demasiado costoso.

Para resolver ambos problemas se puede proceder a buscar una solución del problema que, a cambio de permitir un cierto grado de suboptimalidad, reduzca los costes indicados. Para ello, se pueden seguir dos enfoques diferentes que se analizarán a continuación: simplificar el problema a resolver o simplificar la solución de dicho problema.

### 5.7.1. Simplificación del problema

El problema de optimización a resolver asociado al control predictivo con restricciones no convexas es, como se ha visto en el capítulo 3, el enunciado en (3.2):

$$\mathcal{P}_{N,M}(x): \quad V_{N,M}^{OPT}(x) := \min \tfrac{1}{2}\mathbf{u}^T H \mathbf{u} + \mathbf{u}^T F x,$$

sujeto a:
$u_k \in \bar{\mathbb{U}}$ para $k = 0, \ldots, M-1$,
$u_k = 0$ para $k = M, \ldots, N-1$,
$(\Omega_k x + \Gamma_k \mathbf{u}) \in \bar{\mathbb{X}}$ para $k = 0, \ldots, N$,
$(\Omega_N x + \Gamma_N \mathbf{u}) \in \bar{\mathbb{X}}_f$

Agrupando todas las acciones de control futuras en un vector, el problema se puede reescribir de la forma (3.3):

$$\begin{aligned} \mathcal{P}_{N,M}(x): \quad & V_{N,M}^{OPT}(x) := \min \tfrac{1}{2}\mathbf{u}^T H \mathbf{u} + \mathbf{u}^T F x, \\ & \text{sujeto a:} \\ & (\mathbf{u}, x) \in \bar{\mathbb{T}} \end{aligned}$$

donde:

$$\begin{aligned} \bar{\mathbb{T}} &= \bigcup_{i=0}^{\gamma} T_i \\ T_i &:= \left\{ (\mathbf{u}, x) \in \mathbb{R}^{Mm+n} \,\middle|\, \begin{bmatrix} \Phi_i & \Lambda_i \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ x \end{bmatrix} \leq \Delta_i \right\} \end{aligned}$$

Como se vio en dicho capítulo, los distinto conjuntos $T_i$ son los formados considerando todas las posibles combinaciones de restricciones formando un total de $\gamma = \gamma_u^M \gamma_x^{N-1} \gamma_f$.

Si se analizan las expresiones del coste de los algoritmos propuestos, tanto para la obtención de la acción de control en línea como para el cálculo de la solución explícita del problema fuera de línea, se puede deducir que dependen de manera crítica del parámetro $\gamma$, y que disminuirían considerablemente reduciendo dicho parámetro.

Las dos posibilidades más directas para disminuir $\gamma$ son:

- Reducir el número de regiones convexas admisibles para los conjuntos de restricciones $\bar{\mathbb{X}}$, $\bar{\mathbb{U}}$ o $\bar{\mathbb{X}}_f$: esto implica modificar el enunciado del problema o las garantías de estabilidad, por lo que en principio se descartará.

- Reducir los horizontes de control $M$ o predicción $N$: éstos sí son parámetros de diseño sobre los que se puede actuar. No obstante, dependiendo de la dinámica del sistema a controlar, reducirlos demasiado puede suponer una disminución no aceptable en la calidad de la respuesta que se puede conseguir del sistema.

Si las posibilidades anteriores no son viables, otra opción surge mediante una combinación de ambas, teniendo en cuenta la filosofía del horizonte móvil del control predictivo. La idea es, considerando que sólo se aplican las $m$ primeras acciones de control, aplicar las restricciones exactas a dichas acciones de control pero sustituirlas por un nuevo espacio de restricciones convexo para las acciones de control y/o los estados siguientes. De una manera más general, se aplican las restricciones de manera exacta a $L < M \leq N$ acciones de control, y el nuevo

conjunto de restricciones convexo al resto de acciones de control (5.8).

$$\begin{array}{ll}\mathcal{P}_{N,M}^{ap}(x): & V_{N,M}^{ap}(x) := \min \frac{1}{2}\mathbf{u}^T H\mathbf{u} + \mathbf{u}^T Fx, \\ & \text{sujeto a:} \\ & \quad u_k \in \bar{\mathbb{U}} \text{ para } k = 0, \ldots, L, \\ & \quad u_k \in \bar{\mathbb{U}}_0 \text{ para } k = L+1, \ldots, M-1, \\ & \quad u_k = 0 \text{ para } k = M, \ldots, N-1, \\ & \quad (\Omega_k x + \Gamma_k \mathbf{u}) \in \bar{\mathbb{X}} \text{ para } k = 1, \ldots, L, \\ & \quad (\Omega_k x + \Gamma_k \mathbf{u}) \in \bar{\mathbb{X}}_0 \text{ para } k = L+1, \ldots, N-1, \\ & \quad (\Omega_N x + \Gamma_N \mathbf{u}) \in \bar{\mathbb{X}}_{f0}\end{array} \tag{5.8}$$

donde $\mathbb{U}_0$, $\mathbb{X}_0$ y $\mathbb{X}_{f0}$ son poliedros convexos.

Con esta simplificación, si se agrupan las acciones de control futuras en un vector como se hizo con el problema original, se obtendrá un menor número $\gamma$ de conjuntos $T_i$, en concreto $(\gamma_u \cdot \gamma_x)^L$.

La opción más sencilla para la elección de estas restricciones convexas es buscar poliedros que contengan el origen de sus respectivos espacios y estén, a su vez, contenidos en $\bar{\mathbb{U}}$, $\bar{\mathbb{X}}$ y $\bar{\mathbb{X}}_f$ respectivamente.

Con esta selección, resulta obvio que tras resolver el problema (5.8) se va a obtener una solución igual o peor que la que se obtiene resolviendo el problema original (3.2), por ser las restricciones del nuevo problema más estrictas. No obstante, siempre que el horizonte $L$ al que se aplican las restricciones reales no sea excesivamente inferior a los horizontes de control y predicción, el comportamiento del sistema en bucle cerrado no tiene por qué verse afectado gravemente, puesto que es de esperar que los últimos elementos de la secuencia de control óptima y la trayectoria predicha asociada a ésta se encuentren en un entorno cercano a los respectivos origenes de ambos espacios.

Si esta simplificación se considera demasiado restrictiva, otra opción es tomar como conjuntos convexos las envolventes convexas de los conjuntos de restricciones originales:

$$\begin{array}{ll}\mathcal{P}_{N,M}^{ap}(x): & V_{N,M}^{ap}(x) := \min \frac{1}{2}\mathbf{u}^T H\mathbf{u} + \mathbf{u}^T Fx, \\ & \text{sujeto a:} \\ & \quad u_k \in \bar{\mathbb{U}} \text{ para } k = 0, \ldots, L, \\ & \quad u_k \in \mathbf{conv}(\bar{\mathbb{U}}) \text{ para } k = L+1, \ldots, M-1, \\ & \quad u_k = 0 \text{ para } k = M, \ldots, N-1, \\ & \quad (\Omega_k x + \Gamma_k \mathbf{u}) \in \bar{\mathbb{X}} \text{ para } k = 1, \ldots, L, \\ & \quad (\Omega_k x + \Gamma_k \mathbf{u}) \in \mathbf{conv}(\bar{\mathbb{X}}) \text{ para } k = L+1, \ldots, N-1, \\ & \quad (\Omega_N x + \Gamma_N \mathbf{u}) \in \mathbf{conv}(\bar{\mathbb{X}}_f)\end{array} \tag{5.9}$$

De esta forma, podrán obtenerse acciones de control o trayectorias futuras no factibles, pero las acciones de control que realmente deben aplicarse sí lo serán.

Analizando la formulación de los dos problemas de optimización, (3.2) y (5.9), puede deducirse que $V_{N,M}^{ap}(x) \leq V_{N,M}^{OPT}(x)$ puesto que para el problema aproximado siempre es posible la misma solución que se tiene para el problema original debido a que $\bar{\mathbb{U}} \in \mathbf{conv}(\bar{\mathbb{U}})$, $\bar{\mathbb{X}} \in \mathbf{conv}(\bar{\mathbb{X}})$ y $\bar{\mathbb{X}}_f \in \mathbf{conv}(\bar{\mathbb{X}}_f)$. No

obstante, si se obtiene un coste $V^{ap}_{N,M}(x) \neq V^{OPT}_{N,M}(x)$ es porque alguna de las acciones de control o variables de estado futuras $u_k$ para $k = L+1, \dots M$ solución de $\mathcal{P}^{ap}_{N,M}(x)$ no es factible con las restricciones reales, y por tanto no es válida. Por tanto, aplicar las $L$ primeras acciones de control solución de $\mathcal{P}^{ap}_{N,M}(x)$ supone realmente una estrategia subóptima, puesto que equivale a resolver el problema (5.10).

$$\begin{array}{ll} \mathcal{P}^{sub}_{N,M}(x): & V^{sub}_{N,M}(x) := \min \frac{1}{2}\mathbf{u}^T H \mathbf{u} + \mathbf{u}^T F x, \\ & \text{sujeto a:} \\ & \quad u_k = u^{ap}_k \qquad k = 0 \dots L \\ & \quad u_k \in \bar{\mathbb{U}} \text{ para } k = L+1, \dots, M-1, \\ & \quad u_k = 0 \text{ para } k = M, \dots, N-1, \\ & \quad (\Omega_k x + \Gamma_k \mathbf{u}) \in \bar{\mathbb{X}} \text{ para } k = L+1, \dots, N-1, \\ & \quad (\Omega_N x + \Gamma_N \mathbf{u}) \in \bar{\mathbb{X}}_f \end{array} \tag{5.10}$$

donde $u^{ap}_k$ representa el elemento $k$ de la solución al problema $\mathcal{P}^{ap}_{N,M}(x)$.

Comparando (5.10) con (3.2) queda claro que se cumple $V^{sub}_{N,M}(x) \geq V^{OPT}_{N,M}(x)$ puesto que el problema subóptimo tiene un menor número de grados de libertad, al estar fijadas las $L$ primeras acciones de control.

A continuación, se va a plantear un procedimiento para analizar la suboptimalidad de la solución del problema (5.10) con respecto a la solución óptima de (3.2). Para ello, será necesario calcular dicha solución óptima, por lo que este procedimiento carece de sentido si el objetivo de la simplificación es reducir el coste del algoritmo fuera de línea. No obstante, si el objetivo es la reducción del coste en línea, puede ser de utilidad para conocer el grado de suboptimalidad que se está permitiendo.

En primer lugar, se obtendrán, mediante alguno de los métodos expuestos en el capítulo 4, las soluciones explícitas del problema original (3.2) y el aproximado (5.9) . A continuación, se obtendrá la solución explícita de (5.10). Para ello, en primer lugar, se divide el vector de acciones de control $\mathbf{u}$ de la siguiente forma:

$$\mathbf{u} = \begin{bmatrix} \mathbf{u_1} \\ \mathbf{u_2} \end{bmatrix}$$

donde:

$$\mathbf{u_1} = \begin{bmatrix} u_1 \\ \vdots \\ u_L \end{bmatrix} \quad \mathbf{u_2} = \begin{bmatrix} u_{L+1} \\ \vdots \\ u_M \end{bmatrix}$$

Dado que $\mathbf{u_1}$ es una variable fijada previamente a la resolución del problema (5.10), puede ser tratada como un parámetro. De esta forma, el índice de coste puede reescribirse como:

$$\begin{aligned} V_{N,M}(x) &= \frac{1}{2}\mathbf{u}^T H \mathbf{u} + \mathbf{u}^T F x = \frac{1}{2}\begin{bmatrix} \mathbf{u_1} \\ \mathbf{u_2} \end{bmatrix}^T \begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix} \begin{bmatrix} \mathbf{u_1} \\ \mathbf{u_2} \end{bmatrix} + \begin{bmatrix} \mathbf{u_1} \\ \mathbf{u_2} \end{bmatrix}^T F x = \\ &= \frac{1}{2}\mathbf{u_2}^T H_{22} \mathbf{u_2} + \mathbf{u_2}^T [H_{21} \quad F] \begin{bmatrix} \mathbf{u_1} \\ x \end{bmatrix} + \frac{1}{2}\mathbf{u_1}^T H_{11} \mathbf{u_1} + \mathbf{u_1}^T F x \end{aligned}$$

Por tanto, la solución al problema de optimización (5.10), que fija una expresión concreta de $\mathbf{u_1}$, es un caso particular de la solución de (5.11).

$$\begin{aligned}\mathcal{P}^{sub}_{N,M}(x): \quad & V^{sub}_{N,M}(x) := \min \tfrac{1}{2}\mathbf{u_2}^T H_{22}\mathbf{u_2} + \mathbf{u_2}^T [H_{21} \quad F] \begin{bmatrix}\mathbf{u_1}\\ x\end{bmatrix}, \\ & \text{sujeto a:} \\ & \quad u_k \in \bar{\mathbb{U}} \text{ para } k = L+1,\dots,M, \\ & \quad (\Omega_k x + \Gamma_k \mathbf{u}) \in \bar{\mathbb{X}} \text{ para } k = L+1,\dots,N-1, \\ & \quad (\Omega_N x + \Gamma_N \mathbf{u}) \in \bar{\mathbb{X}}_f\end{aligned} \tag{5.11}$$

El problema (5.11) es también del mismo tipo que el problema original, y por tanto puede resolverse mediante los mismos métodos, obteniéndose una solución explícita expresada en el espacio ampliado de $[\mathbf{u_1} \quad x]$:

$$\mathbf{u_2}^{sub} = F_i^{sub} \begin{bmatrix}\mathbf{u_1}\\ x\end{bmatrix} + G_i^{sub} \quad \text{para } x \in \mathbb{X}_i$$

$$\mathbb{X}_i := x \in \mathbb{R}^n \left| \begin{array}{c} [\mathbf{u_1}^T x^T] M_{ij} \begin{bmatrix}\mathbf{u_1}\\ x\end{bmatrix} + N_{ij} \begin{bmatrix}\mathbf{u_1}\\ x\end{bmatrix} + C_{ij} \le 0 \quad j = 1 \dots q_i \\ L_i \begin{bmatrix}\mathbf{u_1}\\ x\end{bmatrix} + W_i \le 0 \end{array} \right.$$

La solución anterior es válida para cualquier $\mathbf{u_1}$ pero, para el problema (5.10), la expresión de $\mathbf{u_1}$ es conocida y dada por la solución de (5.9):

$$\mathbf{u_1}^{ap} = F_k^{ap} x + G_k^{ap} \quad \text{para } x \in \mathbb{X}_k$$

$$\mathbb{X}_k := x \in \mathbb{R}^n \left| \begin{array}{c} x^T M_{kl} x + N_{kl} x + C_{kl} \le 0 \quad l = 1 \dots q_k \\ L_k x + W_k \le 0 \end{array} \right.$$

A continuación, se comprueba cuáles de los conjuntos $(\mathbb{X}_i, \mathbb{X}_k, \mathbf{u_1}^{ap})$ no están vacíos resolviendo la factibilidad de:

$$\begin{cases} [\mathbf{u_1^{ap}}^T x^T] M_{ij} \begin{bmatrix}\mathbf{u_1^{ap}}\\ x\end{bmatrix} + N_{ij} \begin{bmatrix}\mathbf{u_1^{ap}}\\ x\end{bmatrix} + C_{ij} \le 0 \quad j = 1 \dots q_i \\ L_i \begin{bmatrix}\mathbf{u_1^{ap}}\\ x\end{bmatrix} + W_i \le 0 \\ x^T M_{kl} x + N_{kl} x + C_{kl} \le 0 \quad l = 1 \dots q_k \\ L_k x + W_k \le 0 \\ \mathbf{u_1}^{ap} = F_k^{ap} x + G_k^{ap} \end{cases} \tag{5.12}$$

Como puede observarse, se trata de un problema de factibilidad del mismo tipo que los que han ido resolviéndose anteriormente, y por tanto puede formularse de la misma manera mediante problemas SOS.

Todos aquellas soluciones de los problemas (5.12) que efectivamente resulten factibles, representan las regiones de la partición del espacio de estados solución del problema (5.10), $\mathbb{X}^{sub}$.

Una vez calculada esta solución, el objetivo es comparar el índice de coste que se obtiene con ella con el índice de coste obtenido con la solución óptima del problema, que como se ha visto es inferior.

El índice de coste que, como se ha visto en capítulos anteriores, es cuadrático a tramos, tiene la siguiente expresión para cada una de las dos soluciones a comparar:

$$V_{N,M}^{OPT}(x) = x^T Q_i^{OPT} x + T_i^{OPT} x + V_i^{OPT} \quad \text{para } x \in \mathbb{X}_i^{OPT}$$
$$\mathbb{X}_i^{OPT} := x \in \mathbb{R}^n \left| \begin{array}{c} x^T M_{ij} x + N_{ij} x + C_{ij} \leq 0 \quad j = 1 \dots q_i \\ L_i x + W_i \leq 0 \end{array} \right.$$

$$V_{N,M}^{sub}(x) = x^T Q_k^{sub} x + T_k^{sub} x + V_k^{sub} \quad \text{para } x \in \mathbb{X}_k^{sub}$$
$$\mathbb{X}_k^{sub} := x \in \mathbb{R}^n \left| \begin{array}{c} x^T M_{kl} x + N_{kl} x + C_{kl} \leq 0 \quad l = 1 \dots q_k \\ L_k x + W_k \leq 0 \end{array} \right.$$

Como ya se ha razonado anteriormente, se sabe que para cualquier $x$ se cumplirá $V_{N,M}^{sub}(x) \geq V_{N,M}^{OPT}(x)$, pero lo que interesa ahora es estimar la diferencia entre ambos costes para decidir si la simplificación es adecuada. Para ello, se va a buscar una cota superior sobre la diferencia relativa entre ambos índices de coste, $\frac{V^{sub}(x) - V^{OPT}(x)}{V^{OPT}(x)}$. Teniendo en cuenta que ambos índices de coste están definidos a tramos, la cota superior, $\lambda$, puede obtenerse resolviendo el siguiente problema de optimización:

$$\begin{array}{ll} \min & \lambda \\ \text{sujeto a:} & \\ & \lambda \geq \frac{V_k^{sub}(x) - V_i^{OPT}(x)}{V_i^{OPT}(x)} \quad \forall i, k \\ & x \in \mathbb{X}_i^{OPT} \quad \forall i \\ & x \in \mathbb{X}_k^{sub} \quad \forall k \end{array}$$

De nuevo, el problema anterior puede formularse como un problema de tipo SOS. Una vez resuelto, se tendrá una cota superior del error porcentual máximo que se cometerá resolviendo el problema (5.10) en lugar del (3.2). Si dicho valor se considera aceptable, podrá utilizarse la solución simplificada. En caso contrario, puede repetirse el proceso aumentando el número de acciones de control cuyas restricciones sí son consideradas de manera exacta, $L$.

**Ejemplo 5.2.** *Se desea obtener una solución simplificada al problema propuesto en el ejemplo 4.8, por considerar que la solución obtenida todavía es demasiado compleja para su implementación en línea. Para ello, se plantea aplicar la simplificación* (5.8)*, consistente en sustituir las restricciones originales por subconjuntos poliédricos convexos para los últimos elementos de la secuencia de control. En concreto, se toma una* $L = 2$ *y unas restricciones simplificadas sobre la acción de control* $\mathbb{U}_0 = \mathbb{U}_1$*.*

*En este caso, se tiene un problema definido por* 4 *conjuntos* $T_i$*. Tras la aplicación de la metodología propuesta, se obtiene la partición mostrada en la figura 5.1, definida por 86 regiones, 67 con una única solución, 12 con dos y 7 con tres.*

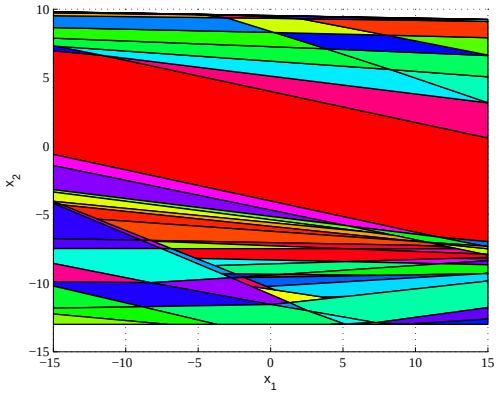


***Figura 5.1:*** Solución final $N = 5$,$L = 2$. Partición del estado.

*Para la obtención de esta solución, se han resuelto 425 problemas de suma de cuadrados y se han empleado un total de 126 segundos en un un Intel core i5 750 a 2.66 GHz.*

*Si se compara esta solución con la obtenida para el caso de $N = 5$ con su resolución exacta (figura 4.17) puede comprobarse como la solución es considerablemente más sencilla (para el caso exacto se tenían 210 regiones, de las que 70 tenían dos soluciones y otras 12 tres) y que además se ha obtenido en mucho menos tiempo en la misma plataforma (1686 segundos frente a 126). A pesar de ello, con esta simplificación la región de factibilidad en la que está definido el controlador no ha disminuido, como puede comprobarse comparando ambas figuras.*

*A efectos de comparación, es conveniente definir un tercer controlador con $N = 2$. Para este caso, el problema tiene una complejidad similar a la del problema con $N = 5$ y $L = 2$, puesto que tiene el mismo número de conjuntos $T_i$, 4. La solución explícita para este controlador es la mostrada en la figura 5.2, y está definida por 47 regiones (31 con una solución, 11 con dos y 4 con tres) obtenidas mediante 196 problemas de suma de cuadrados con un tiempo total de 52 segundos.*

*Puede comprobarse como con este último controlador, la región de factibilidad obtenida sí es menor que la de los dos problemas anteriores.*

*Además, se compara el coste obtenido $J = \sum(y_k^T Q y_k + u_k^T R u_k)$ para las trayectorias con cada uno de los tres controladores desde varios estados iniciales incluidos en las tres regiones de factibilidad hasta alcanzar el origen. La tabla 5.1 muestra el resultado obtenido. Puede comprobarse como, en el peor de los casos, la trayectoria para el controlador con $N = 2$ es hasta un $26{,}8\,\%$ más desfavorable que para $N = 5$. Sin embargo, el controlador $N = 5, L = 2$, que se obtiene con un coste similar, es únicamente un $0{,}4\,\%$ más desfavorable.* ■

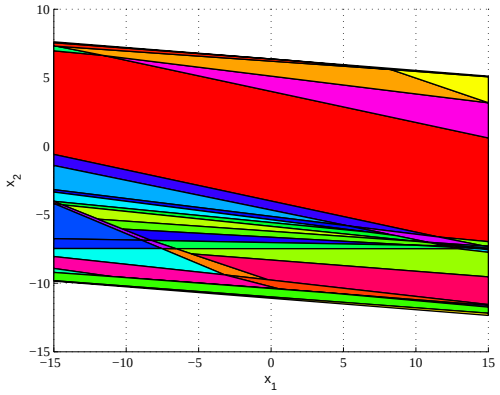


***Figura 5.2:*** Solución final $N = 2$. Partición del estado.

| | N=5 | N=2 | N=5,L=2 |
|---|---|---|---|
| $x_0 = [-15\ 7{,}6]^T$ | 8.98 | 9.60 | 8.98 |
| $x_0 = [15\ 5{,}1]^T$ | 51.47 | 51.47 | 51.47 |
| $x_0 = [15\ -12{,}3]^T$ | 23.58 | 29.89 | 23.66 |
| $x_0 = [-15\ -9{,}8]^T$ | 93.28 | 94.71 | 93.28 |

***Tabla 5.1:*** Índice de coste $J$.

### 5.7.2. Simplificación de la solución

La segunda posibilidad de simplificación planteada consiste en obtener la solución explícita de (3.2) mediante uno de los métodos propuestos en el capítulo 4 y simplificar la partición obtenida. Obviamente, al partirse de la solución explícita exacta, este tipo de simplificación sólo tiene sentido cuando lo que se desea es reducir el coste del algoritmo en línea o el coste de la obtención de éste (obtención del árbol binario) pero, a diferencia de la simplificación del problema propuesta, no es de utilidad cuando lo que es demasiado costoso es la propia obtención de la solución explícita.

La idea fundamental consiste en sustituir curvas cuadráticas, cuyo coste de evaluación es de $n(2n+3)$ operaciones elementales, por hiperplanos, con un coste de $2n+1$ operaciones elementales. Si el algoritmo en línea requiere la evaluación de muchas curvas, el coste en línea puede verse de esta forma considerablemente reducido. No obstante, se pretende que al sustituir las curvas el grado de suboptimalidad de la solución simplificada sea mínimo.

Se partirá de una de las regiones de la solución explícita definida por una serie de desigualdades lineales y cuadráticas:

$$\mathbb{X}_i^{OPT} := x \in \mathbb{R}^n \left| \begin{array}{c} x^T M_{ij} x + N_{ij} x + C_{ij} \leq 0 \quad j = 1 \dots q_i \\ L_i x + W_i \leq 0 \end{array} \right.$$

Renombrando la curva que se desea aproximar como $h(x)$ y el resto de desigual-

dades que forman la región como $f(x)$ y cambiando el signo de las inecuaciones, se puede reescribir la región como:

$$\mathbb{X}_i^{OPT} := x \in \mathbb{R}^n \left| \begin{array}{ll} h(x) \geq 0 & \\ f_i(x) \geq 0 & i = 1 \dots q \end{array} \right.$$

El criterio para realizar la aproximación, como se ha dicho, es que sea lo más próxima posible a la óptima, en la que es necesario dividir la región por la curva cuadrática $h(x)$. En su lugar, se buscará un hiperplano, $t(x)$, con el que sustituirla. Con esta simplificación, siempre que $h(x)$ y $t(x)$ tengan el mismo signo, la acción de control que se aplicará será correcta, y por tanto el error será nulo. Sin embargo, cuando ambos signos difieran, en la solución simplificada se aplicará una acción de control diferente a la óptima, con lo que se cometerá un determinado error. Para cuantificarlo de manera adecuada, se define el error relativo de la siguiente forma:

$$er(x) = \begin{cases} \frac{V_1^{OPT}(x) - V_2^{OPT}(x)}{V_2^{OPT}(x)} & \text{para } V_1^{OPT}(x) > V_2^{OPT}(x) \\ \frac{V_2^{OPT}(x) - V_1^{OPT}(x)}{V_1^{OPT}(x)} & \text{para } V_1^{OPT}(x) < V_2^{OPT}(x) \end{cases}$$

Recordando que , como se vio en el capítulo 4, las curvas que definen las regiones siempre son la resta de dos índices de coste, $h(x) = V_1^{OPT}(x) - V_2^{OPT}(x)$, se puede reescribir el error relativo como:

$$er(x) = \begin{cases} \frac{h(x)}{V_2^{OPT}(x)} & \text{para } h(x) > 0 \\ \frac{-h(x)}{V_1^{OPT}(x)} & \text{para } h(x) < 0 \end{cases}$$

El objetivo es buscar un hiperplano $t(x)$ que minimice el error relativo para todos aquellos $x$ pertenecientes a la región $\mathbb{X}_i$ en los que se sabe que la acción de control a aplicar va a ser subóptima, es decir, aquellos valores de $x$ para los que que el signo de $t(x)$ difiere del de $h(x)$. Esto ocurre en dos subconjuntos diferentes:

$$\left\{ \begin{array}{c} h(x) > 0 \\ t(x) < 0 \\ f_i(x) > 0 \quad \forall i \end{array} \right. \qquad \left\{ \begin{array}{c} h(x) < 0 \\ t(x) > 0 \\ f_i(x) > 0 \quad \forall i \end{array} \right.$$

Una posible forma de plantear la solución del problema, es mediante la aplicación del Positivstellensatz. Se buscará el mínimo $\lambda$ que haga no factibles simultáneamente los dos siguientes conjuntos de desigualdades:

$$\left\{ \begin{array}{c} h(x) > 0 \\ t(x) < 0 \\ f_i(x) > 0 \quad \forall i \\ er(x) > \lambda \end{array} \right. \qquad \left\{ \begin{array}{c} h(x) < 0 \\ t(x) > 0 \\ f_i(x) > 0 \quad \forall i \\ er(x) > \lambda \end{array} \right.$$

Dado que, de la obtención de la solución explícita, se sabe que $h(x)$ pasa por la región $\mathbb{X}_i$, si ambos conjuntos anteriores no son factibles es porque se cumple $er(x) < \lambda$, que sería una cota del error relativo.

Substituyendo $er(x)$ por su expresión correspondiente en cada caso:

$$\begin{cases} h(x) > 0 \\ t(x) < 0 \\ f_i(x) > 0 \quad \forall i \\ h(x) > \lambda V_2^{OPT}(x) \end{cases} \qquad \begin{cases} h(x) < 0 \\ t(x) > 0 \\ f_i(x) > 0 \quad \forall i \\ -h(x) > \lambda V_1^{OPT}(x) \end{cases}$$

El problema de optimización a plantear es:

$$\begin{array}{ll} \min & \lambda \\ \text{sujeto a:} & \\ & \begin{cases} h(x) > 0 \\ t(x) < 0 \\ f_i(x) > 0 \quad \forall i \\ h(x) > \lambda V_2^{OPT}(x) \end{cases} \quad \text{no factible} \\ & \begin{cases} h(x) < 0 \\ t(x) > 0 \\ f_i(x) > 0 \quad \forall i \\ -h(x) > \lambda V_1^{OPT}(x) \end{cases} \quad \text{no factible} \end{array}$$

Como se vio en el capítulo 4, la no factibilidad de un conjunto de desigualdades puede expresarse como la búsqueda de polinomios suma de cuadrados que cumplan la condición del Positivstellensatz:

$$\begin{array}{ll} \min & \lambda \\ \text{sujeto a:} & \\ & -1 = s_0 + \sum_i s_i f_i(x) + s_\lambda(h(x) - \lambda V_2^{OPT}(x)) - s_t t(x) + \\ & + s_h h(x) + \sum_{i,j} s_{ij} f_i(x) f_j(x) + \dots \\ & -1 = s'_0 + \sum_i s'_i f_i(x) + s'_\lambda(-h(x) - \lambda V_1^{OPT}(x)) - s'_h h(x) + \\ & + s'_t t(x) + \sum_{i,j} s'_{ij} f_i(x) f_j(x) + \dots \\ & s_i, s'_i \quad \text{SOS} \end{array} \tag{5.13}$$

El problema (5.13), a diferencia de los problemas SOS enunciados en el capítulo 4, presenta un inconveniente que dificulta su resolución como SDP. Esto es debido a que el problema no es lineal en los parámetros por los productos $s'_t t(x)$ y $s'_\lambda \lambda$,$s_t t(x)$,$s_\lambda \lambda$. Este tipo de restricciones, son lo que se denomina desigualdades bilinieales matriciales (*bilinear matrix inequalities*, BMIs) y requieren de *solvers* específicos diferentes en general a los válidos para la resolución de SDPs.

Los problemas con BMIs son no convexos [VB00] y para su resolución se emplean métodos de *branch and bound* que suelen ser muy costosos computacionalmente [TO95].

Para el problema en cuestión, se intentará evitar la resolución del problema BMI haciendo todos los polinomios enumerados iguales a la unidad. El problema

resultante a resolver en este caso es:

$$\begin{array}{l}\min \quad \lambda \\ \text{sujeto a:} \\ \qquad \begin{array}{l} -1 = s_0 + \sum_i s_i f_i(x) + s_h h(x) - t(x) + (h(x) - \lambda V_2^{OPT}(x)) + \\ + \sum_{i,j} s_{ij} f_i(x) f_j(x) + \dots \\ -1 = s'_0 + \sum_i s'_i f_i(x) - s'_h h(x) + t(x) + (-h(x) - \lambda V_1^{OPT}(x)) + \\ + \sum_{i,j} s'_{ij} f_i(x) f_j(x) + \dots \\ s_i, s'_i \quad \text{SOS} \end{array} \end{array} \tag{5.14}$$

Con esta simplificación, el problema ya puede escribirse como un SDP, y por tanto usar cualquiera de los *solvers* existentes. El principal inconveniente que se tiene resolviendo el problema así es que la cota $\lambda$ que se obtiene puede no ser todo lo buena que podría ser, puesto que se están perdiendo los grados de libertad que dan los polinomios simplificados.

Una vez se tiene esta solución, se puede plantear la resolución iterativa del problema BMI de forma sencilla formulando varios problemas SDP. La idea consiste en tomar, igual que en el caso anterior, $s_\lambda = s'_\lambda = 1$ pero permitir $s_t \neq 1$ y $s'_t \neq 1$. Para obtener un problema que pueda resolverse, hay que tomar un hiperplano con una valor dado $t = t_0$, por ejemplo el polinomio obtenido resolviendo (5.14). Así, habrá que resolver:

$$\begin{array}{l}\min \quad \lambda \\ \text{sujeto a:} \\ \qquad \begin{array}{l} -1 = s_0 + \sum_i s_i f_i(x) + s_h h(x) - s_t t_0(x) + s_\lambda (h(x) - \lambda V_2^{OPT}(x)) + \\ + \sum_{i,j} s_{ij} f_i(x) f_j(x) + \dots \\ -1 = s'_0 + \sum_i s'_i f_i(x) + s'_\lambda (-h(x) - \lambda V_1^{OPT}(x)) - s'_h h(x) + \\ + s'_t t_0(x) + \sum_{i,j} s'_{ij} f_i(x) f_j(x) + \dots \\ s_i, s'_i \quad \text{SOS} \end{array} \end{array}$$

Este problema ya es resoluble mediante SDPs. A continuación, y con los polinomios $s_t = s_{t0}$ y $s'_t = s'_{t0}$ obtenidos, se puede resolver un nuevo SDP en el que éstos pasen a ser fijados y el polinomio $t$ libre:

$$\begin{array}{l}\min \quad \lambda \\ \text{sujeto a:} \\ \qquad \begin{array}{l} -1 = s_0 + \sum_i s_i f_i(x) + s_h h(x) - s_{t0} t(x) + s_\lambda (h(x) - \lambda V_2^{OPT}(x)) + \\ + \sum_{i,j} s_{ij} f_i(x) f_j(x) + \dots \\ -1 = s'_0 + \sum_i s'_i f_i(x) + s'_\lambda (-h(x) - \lambda V_1^{OPT}(x)) - s'_h h(x) + \\ + s'_{t0} t(x) + \sum_{i,j} s'_{ij} f_i(x) f_j(x) + \dots \\ s_i, s'_i \quad \text{SOS} \end{array} \end{array}$$

El procedimiento iterativo que se establece resolviendo estos dos problemas SDP consecutivamente puede continuar hasta que el valor de la cota $\lambda$ no decrezca significativamente.

**Ejemplo 5.3.** *Como ejemplo de la aproximación de la solución, se tomará la intersección de regiones lineales* $\mathbb{X} = \mathbb{X}_{18} \cap \mathbb{X}_{23}$ *del ejemplo 4.6, definida por*

*las siguientes ecuaciones lineales:*

$$\mathbb{X} = \left\{ x \in \mathbb{R}^2 \;\middle|\; \begin{bmatrix} -0{,}29 & -0{,}96 \\ 0{,}11 & 0{,}99 \\ -0{,}11 & -0{,}99 \\ 1{,}0 & 0{,}0 \end{bmatrix} x \leq \begin{bmatrix} 0{,}0043 \\ -1{,}3 \\ 3{,}5 \\ 20{,}0 \end{bmatrix} \right.$$

*Como se vio en dicho ejemplo, en la solución final la región se encuentra dividida por la curva cuadrática:*

$$h(x) = x^T \begin{bmatrix} 0{,}13 & 1{,}1 \\ 1{,}1 & 9{,}6 \end{bmatrix} x + \begin{bmatrix} 0{,}45 & 17{,}0 \end{bmatrix} x + 7{,}9$$

*Resolviendo el problema aproximado* (5.14)*, se obtiene el hiperplano:*

$$t_1(x) = -6{,}996x_1 - 42{,}21x_2 - 67{,}31$$

*Con este hiperplano, se obtiene una cota superior del error relativo de* $\gamma = 0{,}0098 \quad (0{,}98\,\%)$.

*Si, partiendo de este hiperplano, se aplica el procedimiento iterativo descrito, se obtiene un nuevo valor del hiperplano:*

$$t_2(x) = -6{,}579x_1 - 39{,}76x_2 - 63{,}57$$

*La cota del error en este caso es de* $\gamma = 0{,}0061 \quad (0{,}61\,\%)$.

*La figura 5.3 representa la curva* $h(x)$ *y las dos aproximaciones en el poliedro* $\mathbb{X}$*. Como era de esperar a la vista de los resultados de* $\gamma$ *obtenidos, ambas aproximaciones son muy buenas.*

■

## 5.8. Conclusiones

En este capítulo se han introducido algoritmos en línea eficientes para la obtención de la ley de control para el problema del control predictivo con restricciones poliédricas no convexas a partir de una solución explícita obtenida mediante alguno de los métodos planteados en el capítulo 4.

Se han planteado tres algoritmos diferentes, todos ellos basados en la filosofía de los árboles de búsqueda binarios:

- Árboles binarios de los subproblemas convexos y comparación en línea del índice de coste.
- Árbol binario de profundidad mínima: la construcción del árbol requiere la resolución de problemas SOS.

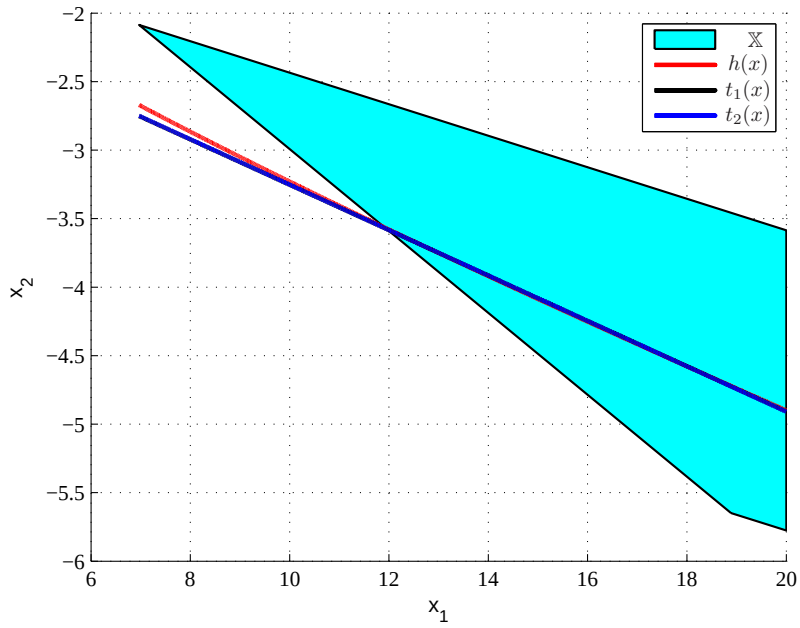


***Figura 5.3:*** Solución final.Partición del estado.

- Árbol binario de profundidad no mínima: árbol binario de una partición lineal y comparación de costes en las regiones en las que sea necesario.

Se ha realizado un estudio y comparación de los tres métodos propuestos, concluyendo que el árbol binario de profundidad no mínima será el más adecuado en la mayoría de ocasiones puesto que con un coste fuera de línea moderado (no requiere la resolución de ningún SOS) se ha demostrado que ofrece un coste en línea menor que el obtenido con los árboles binarios de los subproblemas. En cuanto al árbol de profundidad mínima, dependiendo de la cantidad de desigualdades cuadráticas que sea necesario evaluar puede tener un coste mayor.

El análisis del coste de los algoritmos en línea permite deducir que, para valores de $\gamma$ elevados, puede ser difícil su aplicación para procesos con períodos bajos. Por esta razón, en la última sección del capítulo se proponen algunas simplificaciones que reducen dicho coste a cambio de permitir una cierta suboptimalidad en la solución obtenida. En primer lugar, se han propuesto simplificaciones del problema de optimización, consistentes en reemplazar las restricciones no convexas por su envolvente convexa para instantes posteriores a un determinado horizonte. De esta forma, las primeras acciones de control, que son las únicas que se aplican por la filosofía del horizonte móvil, son factibles, pero las últimas pueden no serlo. A continuación, se plantean también simplificaciones de la solución, consistentes en reemplazar curvas cuadráticas

por hiperplanos, con la consiguiente reducción del coste de evaluación. Para ambos tipos de simplificaciones, que además pueden utilizarse simultáneamente si se desea, se han planteado métodos para el cálculo de la suboptimalidad en qué se incurre.

Capítulo 6

# Estabilidad del control predictivo con restricciones poliédricas no convexas

## 6.1. Introducción

En el capítulo 3 se ha introducido el problema del control predictivo con restricciones poliédricas no convexas sobre acciones de control y variables de estado. En los capítulos 4 y 5 se han desarrollado, respectivamente, algoritmos para el cálculo de la solución explícita al problema y sobre su implementación en línea.

En el presente capítulo, se pretende estudiar la estabilidad del esquema de control predictivo propuesto. Para ello, se particularizan las condiciones del teorema 2.1, válido para cualquier esquema de control óptimo con horizonte finito y estrategia de horizonte móvil.

Además, se propone un algoritmo eficiente para el cálculo de la región terminal óptima para obtener las garantías de estabilidad y se comprueba la estabilidad asintótica del sistema en bucle cerrado.

## 6.2. Condiciones de estabilidad

El objetivo de esta sección es establecer las condiciones suficientes de estabilidad para el problema del control predictivo con restricciones no convexas, definido por el problema de optimización (3.1) y un esquema de horizonte móvil. Para ello, con objeto de simplificar el análisis, se supondrán horizontes de control y predicción iguales ($M = N$). En este caso, el problema de optimización

a resolver queda de la siguiente forma:

$$\mathcal{P}_N(x) : V_N^{OPT}(x) := \min \frac{1}{2} x_N^T P x_N + \frac{1}{2} \sum_{k=0}^{N-1} (x_k^T Q x_k + u_k^T R u_k) \qquad (6.1)$$

sujeto a:
$x_{k+1} = Ax_k + Bu_k$ para $k = 0, \ldots, N-1$,
$x_0 = x$,
$u_k \in \bar{\mathbb{U}}$ para $k = 0, \ldots, N-1$,
$x_k \in \bar{\mathbb{X}}$ para $k = 0, \ldots, N$,
$x_N \in \bar{\mathbb{X}}_f \subset \mathbb{X}$

Puede observarse que dicho problema es una particularización de la estructura general del problema de optimización (2.36) que aparece al analizar la estabilidad del control óptimo con horizonte móvil, enunciado en la sección 2.5:

$$\mathcal{P}_N(x) : \quad V_N^{OPT}(x) := \min V_N(\{x_k\}, \{u_k\}),$$

sujeto a:
$x_{k+1} = f(x_k, u_k)$ para $k = 0, \ldots, N-1$,
$x_0 = x$,
$u_k \in \mathbb{U}$ para $k = 0, \ldots, N-1$,
$x_k \in \mathbb{X}$ para $k = 0, \ldots, N$,
$x_N \in \mathbb{X}_f \subset \mathbb{X}$,
$V_N(\{x_k\}, \{u_k\}) := F(x_N) + \sum_{k=0}^{N-1} L(x_k, u_k)$

Por lo tanto, son aplicables las condiciones suficientes de estabilidad y el teorema 6.1, ya enunciados en dicha sección:

**C1** La ponderación de cada estado $L(x, u)$ en (2.36) satisface $L(0, 0) = 0$ y $L(x, u) \geq \gamma(||x||)$ para todo $x \in \mathbb{S}_N$, $u \in \mathbb{U}$, donde $\gamma : [0, \infty) \to [0, \infty)$ es continua, $\gamma(t) > 0$ para todo $t > 0$ y $\lim_{t \to \infty} \gamma(t) = \infty$.

**C2** La ponderación terminal del estado $F(x)$ en (2.36) satisface $F(0) = 0$, $F(x) \geq 0$ para todo $x \in \mathbb{X}_f$ y cumple la siguiente propiedad: existe una ley de control terminal $\mathcal{K}_f : \mathbb{X}_f \to \mathbb{U}$ tal que $F(f(x, \mathcal{K}_f(x))) - F(x) \leq -L(x, \mathcal{K}_f(x))$ para todo $x \in \mathbb{X}_f$.

**C3** El conjunto $\mathbb{X}_f$ es positivamente invariante para el sistema $x_{i+1} = f(x_i, u_i)$ bajo $\mathcal{K}_f(x)$, es decir, $f(x, \mathcal{K}_f(x)) \in \mathbb{X}_f$ para todo $x \in \mathbb{X}_f$.

**C4** La ley de control terminal $\mathcal{K}_f(x)$ satisface las restricciones del control en $\mathbb{X}_f$, es decir, $\mathcal{K}_f(x) \in \mathbb{U}$ para todo $x \in \mathbb{X}_f$.

**C5** Los conjuntos $\mathbb{U}$ y $\mathbb{X}_f$ contienen el origen de sus respectivos espacios.

**Teorema 6.1.** *Considérese el sistema*

$$x_{i+1} = f(x_i, u_i) \quad \text{para } i \geq 0, \quad f(0, 0) = 0,$$

*controlado por el esquema de horizonte móvil* (2.36)-(2.38) *y supóngase que satisface las condiciones* ***C1-C5*** *enunciadas anteriormente. Entonces:*

1. *El conjunto $\mathbb{S}_N$ de estados iniciales factibles es positivamente invariante para el sistema en bucle cerrado.*

2. *El origen es globalmente atractivo en $\mathbb{S}_N$ para el sistema en bucle cerrado.*

3. *Si, además de **C1**-**C5**, se cumple $0 \in \text{int } \mathbb{S}_N$ y la función de coste $V_N^{OPT}$ es continua en un entorno del origen, entonces el origen es asintóticamente estable en $\mathbb{S}_N$ para el sistema en bucle cerrado.*

4. *Si, además de **C1**-**C5**, se cumple $0 \in \text{int } \mathbb{X}_f$, $\mathbb{S}_N$ es compacta, $\gamma(t) \geq at^\sigma$ en **C1**, $F(x) \leq b||x||^\sigma$ para todo $x \in \mathbb{X}_f$ en **C2**, donde $a > 0$, $b > 0$ y $\sigma > 0$ son constantes reales, y la función de coste $V_N^{OPT}$ es continua en $\mathbb{S}_N$, entonces el origen es exponencialmente estable en $\mathbb{S}_N$ para el sistema en bucle cerrado.*

□

De igual forma que se realizó en la sección 2.5 para sistemas lineales y restricciones convexas, se analizan a continuación las diferentes condiciones de estabilidad:

**C1:** La ponderación de los estados en la función de coste, $L(x,u) = x^TQx + u^TRu$, es la misma que para el caso de restricciones convexas, por lo que el cumplimiento de esta condición puede comprobarse también para una función $\gamma(t) = \lambda_{min}(Q)t^2$, donde $\lambda_{min}(Q)$ es el mínimo valor propio de la matriz $Q$.

**C2:** La función de ponderación terminal y ley de control terminal se escogen en este caso de la misma forma que para el problema con restricciones convexas:

$$F(x_N) = \tfrac{1}{2}x_N^TPx_N$$
$$\mathcal{K}_f = -Kx$$

Donde las matrices $P$ y $K$ puede calcularse resolviendo la ecuación algebraica de Riccati:

$$P = A^TPA + Q - K^T\bar{R}K$$
$$K = \bar{R}^{-1}B^TPA$$
$$\bar{R} = R + B^TPB$$

Por tanto, con esta elección, y teniendo en cuenta que $P$ es definida positiva, se tiene $F(0) = 0$ y $F(x) \geq 0$ para cualquier $x$. Por otra parte:

$$F(f(x,\mathcal{K}_f(x))) - F(x) + L(x,\mathcal{K}_f(x)) =$$
$$= \frac{1}{2}\Big((Ax + B\mathcal{K}_f(x))^TP(Ax + B\mathcal{K}_f(x)) - x^TPx +$$
$$+x^TQx + (\mathcal{K}_f(x))^TR(\mathcal{K}_f(x))\Big)$$

$$
\begin{aligned}
&= \frac{1}{2}x^T \Big( (A-BK)^T P(A-BK) - P + Q + K^T RK \Big) x \\
&= \frac{1}{2}x^T \Big( A^T PA - P + Q + K^T (B^T PB + R)K - 2A^T PBK \Big) x \\
&= \frac{1}{2}x^T \Big( A^T PA - P + Q + (\bar{R}^{-1}B^T PA)^T B^T PA - 2A^T PB\bar{R}^{-1}B^T PA \Big) x \\
&= \frac{1}{2}x^T \Big( A^T PA - P + Q - A^T PB\bar{R}^{-1}B^T PA \Big) x \\
&= \frac{1}{2}x^T \Big( A^T PA - P + Q - A^T PB\bar{R}^{-1}\bar{R}\bar{R}^{-1}B^T PA \Big) x \\
&= \frac{1}{2}x^T \Big( A^T PA - P + Q - K^T \bar{R}K \Big) x = 0
\end{aligned}
$$

**C3 y C4:** Ambas condiciones se satisfacen si se elige como $\mathbb{X}_f$ el máximo conjunto positivamente invariante para el que se satisfacen todas las restricciones en el sistema en bucle cerrado con el controlador terminal, $x_{k+1} = (A - BK)x_k$, es decir el *máximo conjunto positivamente invariante de entrada admisible en bucle cerrado*. No obstante, a diferencia del resto de condiciones, el cálculo de este conjunto invariante difiere en el caso que nos ocupa con el realizado en la sección 2.5, debido a la no convexidad de las regiones de restricciones. Por tanto, el cálculo se analizará en detalle en la sección 6.3.

**C5:** Como en el caso de restricciones convexas, para el cumplimiento de esta condición se asume desde la propia definición del problema que los conjuntos $\mathbb{U}$ y $\mathbb{X}_f$ contienen el origen de sus respectivos espacios.

## 6.3. Obtención de la región terminal

En este sección, se pretende buscar una región terminal $\mathbb{X}_f \subset \bar{\mathbb{X}}$ de forma que las condiciones **C3** y **C4** se satisfagan, es decir, un conjunto invariante en bucle cerrado bajo $\mathcal{K}_f$ para el que la ley de control sea factible en $\bar{\mathbb{U}}$.

Para ello, a continuación se vuelven a enunciar algunas de las definiciones de [Ker00] introducidas en la sección 2.5 haciendo explícito en ellas el conjunto de restricciones sobre las acciones de control considerado:

**Definición 6.1.** Dada una ley de control $u_k = h(x)$, el conjunto de entrada admisible de $\Omega \subset \mathbb{R}^n$ viene dado por

$$\Omega^h(\mathbb{U}) = \{x_k \in \Omega | h(x_k) \in \mathbb{U}\}$$

**Definición 6.2.** El conjunto a un paso en bucle cerrado de entrada admisible $\mathcal{Q}^h(\mathbb{U}, \Omega)$ es el conjunto de estados en $\mathbb{R}^n$ desde los que el sistema $x_{k+1} = f(x_k, u_k)$ en bucle cerrado evoluciona hasta $\Omega$ con una entrada admisible $u_k = h(x) \in \mathbb{U}$, es decir

$$\mathcal{Q}^h(\mathbb{U}, \Omega) = \{x_k \in \mathbb{R}^n | h(x_k) \in \mathbb{U}; f(x_k, h(x_k)) \in \Omega\}$$

**Definición 6.3.** El conjunto controlable a $i$ pasos $\mathbb{K}_i(\mathbb{U},\Omega,\mathbb{T})$ es el máximo conjunto de estados en $\Omega$ para el que existe una secuencia de entradas admisible tal que un conjunto terminal arbitrario $\mathbb{T}\subset\Omega$ sea alcanzado en exactamente $i$ pasos, mientras que el estado se mantiene en $\Omega$ los $i-1$ primeros pasos, es decir

$$\mathbb{K}_i(\mathbb{U},\Omega,\mathbb{T})=\{x_0\in\mathbb{R}^n|\exists\{u_k\in\mathbb{U}\}_0^{i-1};\\ \{x_k\in\Omega\}_0^{i-1},x_i\in\mathbb{T}\}$$

**Definición 6.4.** El conjunto $\mathcal{KO}_i^h(\mathbb{U},\Omega,\mathbb{T})$ para el sistema $x_{k+1}=f(x_k,u_k)$ en bucle cerrado con la ley de control $u_k=h(x)$ se define como $\mathbb{K}_i(\mathbb{U},\Omega^h,\mathbb{T})$ para el sistema $x_{k+1}=f(x_k,h(x_k))$:

$$\mathcal{KO}_i^h(\mathbb{U},\Omega,\mathbb{T})=\{x_0\in\mathbb{R}^n|\{h(x_k)\in\mathbb{U}\}_0^{i-1};\\ \{x_k\in\Omega\}_0^{i-1},x_i\in\mathbb{T}\}$$

Tal y como se ha comentado anteriormente, cuanto mayor sea la región terminal $\mathbb{X}_f$, mayor será el conjunto de estados iniciales factibles $\mathbb{S}_N$ para el controlador predictivo. Por tanto, la región terminal más conveniente es el máximo conjunto positivamente invariante de entrada admisible en bucle cerrado con el controlador terminal $\mathcal{K}_f=-Kx$:

$$\mathcal{O}_\infty^{\mathcal{K}_f}(\bar{\mathbb{U}},\bar{\mathbb{X}})=\{x\in\mathbb{R}^n|K(A-BK)^kx\in\bar{\mathbb{U}},(A-BK)^kx\in\bar{\mathbb{X}}\text{ para }k=0,1,\ldots\}$$

Para el cálculo de $\mathcal{O}_\infty^{\mathcal{K}_f}(\bar{\mathbb{U}},\bar{\mathbb{X}})$, la primera opción a considerar es seguir el procedimiento descrito en el capítulo 2, aprovechando la igualdad $\mathcal{O}_\infty^{\mathcal{K}_f}(\mathbb{U},\Omega)=\mathbb{K}_\infty(\mathbb{U},\Omega^{\mathcal{K}_f},\Omega^{\mathcal{K}_f})=\mathcal{KO}_\infty^{\mathcal{K}_f}(\mathbb{U},\Omega,\Omega^{\mathcal{K}_f})$ y aplicando el algoritmo 6.1:

**Algoritmo 6.1** Cálculo de conjunto controlable en $N$ pasos $\mathbb{K}_N(\mathbb{U},\Omega,\mathbb{T})$

1. Hacer $i=0$ y $\mathbb{K}_0(\mathbb{U},\Omega,\mathbb{T})=\mathbb{T}$
2. Mientras $i<N$:
   *a*) $\mathbb{K}_{i+1}(\mathbb{U},\Omega,\mathbb{T})=\mathcal{Q}\left(\mathbb{U},\mathbb{K}_i(\mathbb{U},\Omega,\mathbb{T})\right)\cap\Omega$
   *b*) Si $\mathbb{K}_{i+1}(\mathbb{U},\Omega,\mathbb{T})=\mathbb{K}_i(\mathbb{U},\Omega,\mathbb{T})$, finalizar el algoritmo y $\mathbb{K}_N(\mathbb{U},\Omega,\mathbb{T})=\mathbb{K}_\infty(\mathbb{U},\Omega,\mathbb{T})=\mathbb{K}_i(\mathbb{U},\Omega,\mathbb{T})$.
   *c*) i=i+1

La principal dificultad para la ejecución del algoritmo 6.1 estriba en el cálculo de los conjuntos a un paso en bucle cerrado $\mathcal{Q}^{\mathcal{K}_f}(\bar{\mathbb{U}},\bar{\mathbb{X}})$ puesto que éstos están definidos sobre conjuntos no convexos:

$$\bar{\mathbb{U}}=\bigcup_i^{\gamma_u}\mathbb{U}_i\quad\bar{\mathbb{X}}=\bigcup_j^{\gamma_x}\mathbb{X}_j \tag{6.3}$$

Para realizar dicho cálculo, será de utilidad la siguiente propiedad [Ker00]:

**Proposición 6.2.** *Si $\Omega$ viene dado por la unión*

$$\Omega=\bigcup_i\Omega_i,$$

*entonces*

$$\mathcal{Q}(\mathbb{U},\Omega)=\bigcup_i \mathcal{Q}(\mathbb{U},\Omega_i)$$

□

Por otra parte, puede formularse la siguiente proposición análoga a la anterior:

**Proposición 6.3.** *Si* $\mathbb{U}$ *viene dado por la unión*

$$\mathbb{U}=\bigcup_i \mathbb{U}_i,$$

*entonces*

$$\mathcal{Q}(\mathbb{U},\Omega)=\bigcup_i \mathcal{Q}(\mathbb{U}_i,\Omega)$$

□

**Prueba.** Si $x_k \in \mathcal{Q}(\mathbb{U},\Omega)$, por definición, existe un $u_k \in \mathbb{U}$ tal que $x_{k+1} \in \Omega$. Pero si $u_k \in \mathbb{U}$, se tiene que $u_k \in \mathbb{U}_i$ para uno o más $i$, por lo que $x_k \in \mathcal{Q}(\mathbb{U}_i,\Omega)$, lo que implica $x_k \in \bigcup_i \mathcal{Q}(\mathbb{U}_i,\Omega)$. Con esto queda probado $\mathcal{Q}(\mathbb{U},\Omega) \subseteq \bigcup_i \mathcal{Q}(\mathbb{U}_i,\Omega)$. En sentido contrario, si $x_k \in \bigcup_i \mathcal{Q}(\mathbb{U}_i,\Omega)$, para algún $i$ se cumple $x_k \in \mathcal{Q}(\mathbb{U}_i,\Omega)$. Aplicando la definición del conjunto a un paso, existe algún $u_k \in \mathbb{U}_i$ tal que $x_{k+1} \in \Omega$. Pero cualquier $u_k \in \mathbb{U}_i$ cumple $u_k \in \mathbb{U}$, por lo que $x_k \in \mathcal{Q}(\mathbb{U},\Omega)$. De esta forma, queda probado $\bigcup_i \mathcal{Q}(\mathbb{U}_i,\Omega) \subseteq \mathcal{Q}(\mathbb{U},\Omega)$ y por lo tanto $\mathcal{Q}(\mathbb{U},\Omega)=\bigcup_i \mathcal{Q}(\mathbb{U}_i,\Omega)$. ■

Aplicando ambas proposiciones a nuestro problema particular, se tiene:

$$\mathcal{Q}^{\mathcal{K}_f}(\bar{\mathbb{U}},\bar{\mathbb{X}})=\bigcup_i\bigcup_j \mathcal{Q}^{\mathcal{K}_f}(\mathbb{U}_i,\mathbb{X}_j) \tag{6.4}$$

Empleando la ecuación (6.4) en el paso 2.*a* del algoritmo 6.1, se tiene un procedimiento completo para la obtención de $\mathcal{O}_\infty^{\mathcal{K}_f}(\bar{\mathbb{U}},\bar{\mathbb{X}})$. No obstante, este algoritmo tiene un inconveniente importante. Esto es debido a que, en cada paso del algoritmo, es necesario calcular el conjunto a un paso que lleva a otro conjunto a un paso, $\mathcal{Q}^{\mathcal{K}_f}(\mathbb{U},\mathcal{Q}^{\mathcal{K}_f}(\mathbb{U},\mathbb{X}))$. Como el conjunto a un paso (6.4) es en general no convexo, el número de conjuntos no convexos puede crecer de manera exponencial en cada paso. Este hecho, además de hacer el algoritmo computacionalmente más costoso, implica que la región terminal $\mathbb{X}_f = \mathcal{O}_\infty^{\mathcal{K}_f}(\bar{\mathbb{U}},\bar{\mathbb{X}})$ quedará definida como la unión de número innecesariamente alto de poliedros convexos. Esto implicará o bien un incremento del coste del algoritmo en línea, o bien la necesidad de utilizar algoritmos de unión de regiones como los introducidos en el capítulo 4, y por tanto un mayor coste fuera de línea.

Con objeto de diseñar un algoritmo más eficiente, se introducen las siguientes nuevas definiciones y la proposición 6.4.

**Definición 6.5.** El *conjunto a i pasos en bucle cerrado de entrada admisible* $\mathcal{Q}_i^h(\mathbb{U},\Omega,\mathbb{T})$ es el conjunto de estados para los que la trayectoria en bucle cerrado dada por $x_{k+1}=f(x_k,h(x_k))$ lleva al estado al conjunto $\mathbb{T}$ en $i$ pasos o menos manteniendo en todo momento la trayectoria de los estados y las acciones de control $u_k=h(x_k)$ dentro de, respectivamente, $\Omega$ y $\mathbb{U}$:

$$\mathcal{Q}_i^h(\mathbb{U},\Omega,\mathbb{T})=\{x_0\in\mathbb{R}^n|\exists N\leq i:\\ \{u_k=h(x_k)\in\mathbb{U}\}_0^{N-1},\{x_k\in\Omega\}_0^{N-1},x_N\in\mathbb{T}\}$$

Este conjunto puede calcularse obteniendo los conjuntos que llevan al sistema en bucle cerrado a $\mathbb{T}$ en exactamente $j$ pasos, es decir, $\mathcal{KO}_j^h(\mathbb{U},\Omega,\mathbb{T})$:

$$\mathcal{Q}_i^h(\mathbb{U},\Omega,\mathbb{T})=\bigcup_{j=1}^{i}\mathcal{KO}_j^h(\mathbb{U},\Omega,\mathbb{T}) \tag{6.5}$$

**Definición 6.6.** El conjunto a $i$ pasos en bucle cerrado de entrada admisible para $i=\infty$ es el *máximo conjunto en bucle cerrado de entrada admisible* $\mathcal{Q}_\infty^h(\mathbb{U},\Omega,\mathbb{T})$ y se dice que está finitamente determinado si existe algún $i^*\in\mathbb{N}$ para el que $\mathcal{Q}_\infty^h(\mathbb{U},\Omega,\mathbb{T})=\mathcal{Q}_{i^*}^h(\mathbb{U},\Omega,\mathbb{T})$ □

Nótese que en ningún momento se asume que el conjunto objetivo $\mathbb{T}$ es invariante, por tanto una condición inicial $x_0\in\mathcal{Q}_\infty^h(\mathbb{U},\Omega,\mathbb{T})$ llevará al sistema en bucle cerrado a $\mathbb{T}$ en un determinado número de pasos, pero se desconoce si la trayectoria del sistema permanecerá en dicho conjunto. Si $\mathbb{T}$ se define como un subconjunto del espacio de estados para el que no se satisfacen algunas de las restricciones del problema, el conjunto $\mathcal{Q}_\infty^h(\mathbb{U},\Omega,\mathbb{T})$ representa los estados que violan dichas restricciones en algún punto de su trayectoria futura.

Para diseñar un procedimiento para el cálculo de $\mathcal{Q}_\infty^h(\mathbb{U},\Omega,\mathbb{T})$ se formula la siguiente proposición:

**Proposición 6.4.** *Si el sistema en bucle cerrado* $x_{k+1}=f(x_k,h(x_k))$ *es asintóticamente estable y* $0\notin\mathbb{T}$*, existe algún* $i^*\in\mathbb{N}$ *tal que* $\mathcal{KO}_i^h(\mathbb{U},\Omega,\mathbb{T})=\emptyset$ $\forall i\geq i^*$. □

**Prueba.** Si $0\notin\mathbb{T}$, existe algún $\epsilon>0$ tal que si $||x||<\epsilon$, entonces $x\notin\mathbb{T}$. Por otro lado, si el sistema en bucle cerrado es asintóticamente estable, existe algún $i^*\in\mathbb{N}$ tal que $||x_i||<\epsilon\ \forall i\geq i^*$. Por tanto, para $i\geq i^*$ $||x_i||\notin\mathbb{T}$ y $\mathcal{KO}_i^h(\mathbb{U},\Omega,\mathbb{T})=\emptyset$. ■

**Observación 6.1.** Nótese que si el sistema en bucle cerrado es asintóticamente estable y $0\notin\mathbb{T}$, de la proposición 6.4 y (6.5), $\mathcal{Q}_\infty^h(\mathbb{U},\Omega,\mathbb{T})$ está finitamente determinado. □

**Observación 6.2.** Para el sistema lineal $x_{k+1}=Ax_k+Bu_k$, si la ley de control se escoge según (2.42) $u_k=\mathcal{K}_f(x)=-Kx$, el sistema en bucle cerrado es por definición asintóticamente estable. □

La proposición y definiciones anteriores pueden utilizarse para diseñar un algoritmo diferente para el cálculo de $\mathcal{O}_\infty^{\mathcal{K}_f}(\bar{\mathbb{U}},\bar{\mathbb{X}})$.

El procedimiento comienza calculando las envolventes convexas de los conjuntos de restricciones $\hat{\mathbb{U}} = \mathbf{conv}(\bar{\mathbb{U}})$ y $\hat{\mathbb{X}} = \mathbf{conv}(\bar{\mathbb{X}})$ y el máximo conjunto admisible $\mathcal{O}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}})$. Dado que $\hat{\mathbb{U}}$ y $\hat{\mathbb{X}}$ son convexos, este conjunto se puede calcular de manera sencilla mediante el algoritmo 6.1.
Puesto que las restricciones originales están incluidas en sus respectivas envolventes, por definición se tiene $\mathcal{O}_\infty^{\mathcal{K}_f}(\bar{\mathbb{U}}, \bar{\mathbb{X}}) \subset \mathcal{O}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}})$.
Por tanto, la idea para el algoritmo es partir de este conjunto e ir eliminando todos los estados para los que la trayectoria en bucle cerrado viole en algún instante futuro alguna restricción (sobre estados o entradas). Este conjunto de estados que deben eliminarse se representa como $\mathbb{X}_f^*$.
Para calcularlo, en primer lugar se obtienen los complementos de los conjuntos de restricciones en sus respectivas envolventes:

$$\hat{\mathbb{U}} \backslash \bar{\mathbb{U}} = \bigcup_i \mathbb{U}_i^* \qquad \hat{\mathbb{X}} \backslash \bar{\mathbb{X}} = \bigcup_j \mathbb{X}_j^*.$$

En muchos casos, como los problemas de evitación de obstáculos, no es necesario calcular las envolventes y los conjuntos complementarios, puesto que las propias restricciones del problema están definidas como un conjunto convexo con algunos subconjuntos prohibidos en su interior.

Nótese que, como $\bar{\mathbb{U}}$ y $\bar{\mathbb{X}}$ por definición contienen el origen de sus respectivos espacios, éste no se encuentra en sus complementos:

$$0 \notin \mathbb{U}_i^*, \quad \forall i$$
$$0 \notin \mathbb{X}_j^*, \quad \forall j$$

Es evidente que, para todos los $\mathbb{U}_i^*$, cualquier estado para el que se cumpla $u_k = \mathcal{K}_f(x_k) \in \mathbb{U}_i^*$ debe ser eliminado. De las definiciones de conjuntos dadas anteriormente, puede deducirse que estos estados se representan como $\mathcal{T}_i = \mathcal{Q}^{\mathcal{K}_f}(\mathbb{U}_i^*, \mathcal{O}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}}))$. Además, cualquier estado a partir del cual la trayectoria del sistema en bucle cerrado lleve al conjunto anterior en cualquier número de pasos, también debe ser eliminado. Estos estados pueden representarse como $\mathcal{Q}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}}, \mathcal{T}_i)$.

Puesto que $\mathbb{U}_i^*$ y $\mathcal{O}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}})$ son convexos, $\mathcal{T}_i$ también lo es, y puede calcularse de manera sencilla aplicando la definición de conjunto a un paso en bucle cerrado. Por otro lado, $\mathcal{Q}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}}, \mathcal{T}_i)$ también está definido a partir de conjuntos convexos. Además, como se cumple $0 \notin \mathbb{U}_i^*$, puede aplicarse la proposición 6.4 y el conjunto está finitamente determinado.

De manera análoga, cualquier estado en el interior de de algún $\mathbb{X}_j^*$ debe ser eliminado: $\mathcal{D}_j = \mathcal{O}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}}) \cap \mathbb{X}_j^*$. De nuevo, deben eliminarse también todos los estados que lleven en su trayectoria futura al conjunto anterior: $\mathcal{Q}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}}, \mathcal{D}_j)$. Como en el caso anterior, puesto que $0 \notin \mathbb{X}_j^*$, el conjunto está finitamente determinado por la aplicación de la proposición 6.4.

Por tanto, el conjunto de estados $\mathbb{X}_f^*$ que debe ser eliminado de $\mathcal{O}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}})$

es:

$$\mathbb{X}_f^* = \bigcup_i (\mathcal{T}_i \cup \mathcal{Q}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}}, \mathcal{T}_i)) \cup \bigcup_j (\mathcal{D}_j \cup \mathcal{Q}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}}, \mathcal{D}_j)) \tag{6.6}$$

Por último, el máximo conjunto positivamente invariante de entrada admisible puede calcularse como:

$$\mathcal{O}_\infty^{\mathcal{K}_f}(\bar{\mathbb{U}}, \bar{\mathbb{X}}) = \mathcal{O}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}}) \backslash \mathbb{X}_f^*$$

El algoritmo 6.2 resume el procedimiento descrito.

**Algoritmo 6.2** Cálculo de $\mathcal{O}_\infty^{\mathcal{K}_f}(\bar{\mathbb{U}}, \bar{\mathbb{X}})$

1. Inicializar el conjunto de estados a eliminar, $\mathbb{X}_f^* = \emptyset$.
2. Calcular la envolvente convexa de cada conjunto de restricciones: $\hat{\mathbb{U}} = \mathbf{conv}(\bar{\mathbb{U}})$, $\hat{\mathbb{X}} = \mathbf{conv}(\bar{\mathbb{X}})$.
3. Calcular el máximo conjunto invariante de entrada admisible para esas restricciones: $\mathcal{O}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}})$.
4. Calcular los complementos de $\bar{\mathbb{U}}$ y $\bar{\mathbb{X}}$ en sus envolventes respectivas: $\hat{\mathbb{U}} \backslash \bar{\mathbb{U}} = \bigcup_i \mathbb{U}_i^*$, $\hat{\mathbb{X}} \backslash \bar{\mathbb{X}} = \bigcup_j \mathbb{X}_j^*$.
5. Para cada $\mathbb{U}_i^*$:
   *a*) Obtener el subconjunto de $\mathcal{O}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}})$ tal que $u_k = \mathcal{K}_f(x_k) \in \mathbb{U}_i^*$: $\mathcal{T}_i = \mathcal{Q}^{\mathcal{K}_f}(\mathbb{U}_i^*, \mathcal{O}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}}))$
   *b*) Calcular el máximo conjunto en bucle cerrado de entrada admisible: $\mathcal{Q}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}}, \mathcal{T}_i)$
   *c*) Añadir ambos conjuntos a $\mathbb{X}_f^*$: $\mathbb{X}_f^* = \mathbb{X}_f^* \cup \mathcal{T}_i \cup \mathcal{Q}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}}, \mathcal{T}_i)$
6. Para cada $\mathbb{X}_j^*$:
   *a*) Obtener el subconjunto de $\mathcal{O}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}})$ tal que $x_k \in \mathbb{X}_j^*$: $\mathcal{D}_j = \mathcal{O}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}}) \cap \mathbb{X}_j^*$.
   *b*) Calcular el máximo conjunto en bucle cerrado de entrada admisible: $\mathcal{Q}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}}, \mathcal{D}_j)$
   *c*) Añadir ambos conjunto a $\mathbb{X}_f^*$: $\mathbb{X}_f^* = \mathbb{X}_f^* \cup \mathcal{D}_j \cup \mathcal{Q}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}}, \mathcal{D}_j)$
7. Calcular el conjunto final como:

$$\mathcal{O}_\infty^{\mathcal{K}_f}(\bar{\mathbb{U}}, \bar{\mathbb{X}}) = \mathcal{O}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}}) \backslash \mathbb{X}_f^*$$

### 6.3.1. Eficiencia de los algoritmos

Como se ha comentado anteriormente, el algoritmo 6.1 tiene el inconveniente fundamental del crecimiento exponencial del número de regiones que lo definen en el peor caso. Este crecimiento depende del número de iteraciones que el algoritmo necesita para converger, $i^*$, que a su vez depende de la dinámica del sistema en bucle cerrado.

Si consideramos las restricciones no convexas sobre las entradas y los estados (6.3), el algoritmo 6.1 comienza haciendo

$$\mathbb{K}_0(\bar{\mathbb{U}}, \bar{\mathbb{X}}^{\mathcal{K}_f}, \bar{\mathbb{X}}^{\mathcal{K}_f}) = \bar{\mathbb{X}}^{\mathcal{K}_f} = \bigcup_{k=0}^{\gamma_x} \mathbb{X}_k$$

A continuación, para una iteración dada, en el paso 2.a es necesario calcular el conjunto:

$$\mathbb{K}_{i+1}(\bar{\mathbb{U}}, \bar{\mathbb{X}}^{\mathcal{K}_f}, \bar{\mathbb{X}}^{\mathcal{K}_f}) = \mathcal{Q}\left(\bar{\mathbb{U}}, \mathbb{K}_i(\bar{\mathbb{U}}, \bar{\mathbb{X}}^{\mathcal{K}_f}, \bar{\mathbb{X}}^{\mathcal{K}_f})\right) \cap \bar{\mathbb{X}}^{\mathcal{K}_f}$$

Aplicando (6.4), se tiene:

$$\begin{aligned}\mathbb{K}_{i+1}(\bar{\mathbb{U}}, \bar{\mathbb{X}}^{\mathcal{K}_f}, \bar{\mathbb{X}}^{\mathcal{K}_f}) &= \left(\bigcup_j^{\gamma_u} \bigcup_k^{\gamma_{\mathbb{K}i}} \mathcal{Q}^{\mathcal{K}_f}(\mathbb{U}_j, \mathbb{K}_{ik}(\bar{\mathbb{U}}, \bar{\mathbb{X}}^{\mathcal{K}_f}, \bar{\mathbb{X}}^{\mathcal{K}_f}))\right) \cap \left(\bigcup_l^{\gamma_x} \mathbb{X}_l\right) \\ &= \bigcup_j^{\gamma_u} \bigcup_k^{\gamma_{\mathbb{K}i}} \bigcup_l^{\gamma_x} \left(\mathcal{Q}^{\mathcal{K}_f}(\mathbb{U}_j, \mathbb{K}_{ik}(\bar{\mathbb{U}}, \bar{\mathbb{X}}^{\mathcal{K}_f}, \bar{\mathbb{X}}^{\mathcal{K}_f})) \cap \mathbb{X}_l\right)\end{aligned}$$

donde $\gamma_{\mathbb{K}_i}$ representa el número de poliedros convexos cuya unión define $\mathbb{K}_i(\bar{\mathbb{U}}, \bar{\mathbb{X}}^{\mathcal{K}_f}, \bar{\mathbb{X}}^{\mathcal{K}_f})$.

Para el peor caso, ningún subconjunto de los poliedros que define $\mathbb{K}_{i+1}$ puede unirse entre sí para formar un nuevo poliedro convexo y, por tanto, este conjunto está definido por $\gamma_u \cdot \gamma_x \cdot \gamma_{\mathbb{K}i}$ regiones.

Considerando el escenario en que esta situación de peor caso se produce a cada iteración del algoritmo, en cada paso se tiene un conjunto definido por el siguiente número de poliedros convexos:

$$\gamma_{\mathbb{K}i} = \gamma_{\mathbb{K}0} \cdot (\gamma_u \gamma_x)^i = \gamma_x (\gamma_u \gamma_x)^i$$

Por tanto, el número de conjuntos que define $\mathcal{O}_\infty^{\mathcal{K}_f}(\bar{\mathbb{U}}, \bar{\mathbb{X}})$ en el peor caso viene dado por

$$\gamma_{\mathbb{K}i^*} = \gamma_x (\gamma_u \gamma_x)^{i^*}$$

que es claramente exponencial en $i^*$ y polinómico en $\gamma_u$ y $\gamma_x$.

Por otro lado, el algoritmo 6.2 requiere el cómputo del conjunto $\mathbb{X}_f^*$. Por tanto, la eficiencia del algoritmo se analizará en términos del coste de cálculo de dicho conjunto.

Tomando los complementos de los conjuntos de restricciones, $\mathbb{U}^* = \bigcup_i^{\gamma_u^*} \mathbb{U}_i^*$ y $\mathbb{X}^* = \bigcup_j^{\gamma_x^*} \mathbb{X}_j^*$, para cada uno de los poliedros convexos que los definen es necesario obtener los conjuntos $\mathcal{Q}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}}, \mathcal{T}_i)$ o $\mathcal{Q}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}}, \mathcal{D}_j)$.

Aplicando (6.5) dichos conjuntos pueden escribirse como (6.7) y la expresión análoga para $\mathcal{D}_j$:

$$\mathcal{Q}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}}, \mathcal{T}_i) = \bigcup_{j=1}^{i_i^*} \mathbb{K}_j(\hat{\mathbb{U}}, \hat{\mathbb{X}}^{\mathcal{K}_f}, \mathcal{T}_i^{\mathcal{K}_f}) \tag{6.7}$$

donde $i_i^*$ depende de la dinámica del sistema en bucle cerrado.

Para el cálculo de cada uno de estos $\mathbb{K}_j$ es necesario aplicar el algoritmo 6.1. No obstante, a diferencia del caso anterior, como $\hat{\mathbb{U}}$, $\hat{\mathbb{X}}$ y $\mathcal{T}_i$ (o $\mathcal{D}_j$) son conjuntos convexos, $\mathbb{K}_j$ es un único poliedro convexo. Por tanto, $\mathcal{Q}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}}, \mathcal{T}_i)$ ($\mathcal{Q}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}}, \mathcal{D}_j)$) está formado en el peor caso por la unión no convexa de $i_i^*$ ($i_j^*$) poliedros convexos.

De esta forma, observando (6.6) y considerando que, como se ha dicho, cada $\mathcal{T}_i$ y $\mathcal{D}_j$ está definido por un único poliedro convexo, el número total de poliedros convexos que forma $\mathbb{X}_f^*$ en el peor caso puede calcularse como:

$$\gamma_{xf}^* = \sum_{i=1}^{\gamma_u^*}(1 + i_i^*) + \sum_{j=1}^{\gamma_x^*}(1 + i_j^*) \leq (\gamma_u^* + \gamma_x^*)(2 + i_{max}^*)$$

donde $i_{max}^*$ es el máximo valor entre todos los $i_i^*$ y $i_j^*$. Puede observarse que $\gamma_{xf}^*$ está acotado por una expresión lineal en $i_{max}^*$, $\gamma_u^*$ y $\gamma_x^*$.

El número de poliedros cuya unión define $\mathcal{O}_\infty^{\mathcal{K}_f}(\bar{\mathbb{U}}, \bar{\mathbb{X}})$ cuando se calcula mediante el algoritmo 6.2 no es $\gamma_{xf}^*$ sino un número relacionado con éste pero dependiente de cada problema particular. No obstante, en general es del mismo orden que $\gamma_{xf}^*$.

Por tanto, en un elevado número de problemas, el algoritmo 6.2 será más eficiente que el 6.1, tanto en términos del número de conjuntos convexos que definen la región final como del tiempo de cómputo para obtenerlo.

Además, si el número final de regiones convexas es minimizado para ambos algoritmo aplicando la metodología descrita en la sección 2.6.2, el algoritmo 6.2 es todavía más eficiente, puesto que en dicha metodología se hace uso de la envolvente y los complementos, conjuntos que ya han sido calculados para el algoritmo mencionado, pero no para el algoritmo 2.6.

**Ejemplo 6.1.** *Se desea obtener la región terminal que garantice estabilidad para el sistema de dos entradas, dos salidas y tres estados que tiene como matriz de transferencia*

$$G(s) = \begin{bmatrix} \frac{5}{s^2+5} & 0 \\ 0 & \frac{2}{s+2} \end{bmatrix}$$

*discretizado con un período de muestreo de* $T_s = 0{,}2$.

*Las restricciones a la entrada no convexas vienen dadas por la unión de dos conjuntos convexos, como muestra la figura 6.1:*

$$\bar{\mathbb{U}} = \mathbb{U}_1 \bigcup \mathbb{U}_2$$
$$\mathbb{U}_1 \equiv \left\{ \begin{array}{c} -2 \leq u_1 \leq 1 \\ -1 \leq u_2 \leq 2 \end{array} \right\}$$
$$\mathbb{U}_2 \equiv \left\{ \begin{array}{c} 1 \leq u_1 \leq 3 \\ -2 \leq u_2 \leq 3 \end{array} \right\}$$

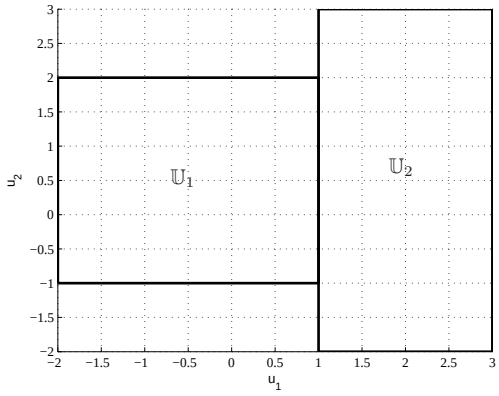


***Figura 6.1:*** Restricciones sobre las acciones de control.

*Como controlador terminal, se escoge el controlador óptimo LQR sin restricciones obtenido para unas matrices de ponderación de los estados $Q = C^T C$ y de las entradas $R = I$: $\mathcal{K}_f(x) = -Kx$.*

*Las figuras 6.2 y 6.3 representan la región $\mathcal{O}_\infty^{\mathcal{K}_f}(\bar{\mathbb{U}}, \bar{\mathbb{X}})$ obtenida aplicando los algoritmos 6.1 y 6.2. Puede comprobarse que la región total no convexa obtenida es la misma mediante los dos métodos aunque las regiones convexas cuya unión forma $\mathcal{O}_\infty^{\mathcal{K}_f}(\bar{\mathbb{U}}, \bar{\mathbb{X}})$ son diferentes.* ■

Como muestra el ejemplo anterior, a pesar de que la región $\mathcal{O}_\infty^{\mathcal{K}_f}(\bar{\mathbb{U}}, \bar{\mathbb{X}})$ obtenida con los dos métodos propuestos es la misma, las regiones convexas cuya unión la define son diferentes en ambos. Este hecho cobra especial importancia al recordar que la región que se está calculando va a ser utilizada como región terminal del control predictivo con objeto de garantizar su estabilidad.

Como se vio en el capítulo 3, el número de conjuntos de restricciones convexos cuya unión define la región terminal afecta directamente al número de conjuntos de restricciones $T_i$, y por tanto a $\gamma$, parámetro fundamental, como se ha visto, para determinar el coste computacional de los algoritmos en línea y fuera de línea. Por tanto, interesa reducir al máximo el número de regiones convexas que forma $\mathbb{X}_f = \mathcal{O}_\infty^{\mathcal{K}_f}(\bar{\mathbb{U}}, \bar{\mathbb{X}})$. Para ello, se aplicarán los algoritmos de

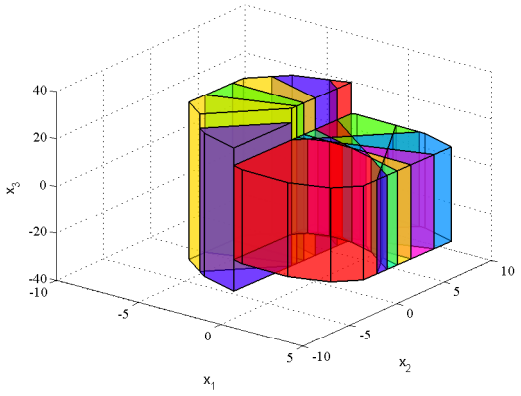


***Figura 6.2:*** $\mathcal{O}_{\infty}^{\mathcal{K}_f}(\bar{\mathbb{U}}, \bar{\mathbb{X}})$ mediante algoritmo 6.1.

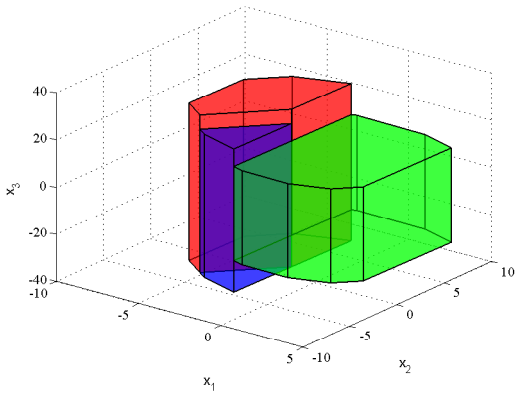


***Figura 6.3:*** $\mathcal{O}_{\infty}^{\mathcal{K}_f}(\bar{\mathbb{U}}, \bar{\mathbb{X}})$ mediante algoritmo 6.2.

reducción de la complejidad de particiones poliédricas de [GTM08], ya tratados en la sección 4.2.4.

**Ejemplo 6.2.** *Para el sistema del ejemplo 6.1 se calcula el conjunto* $\mathcal{O}_{\infty}^{\mathcal{K}_f}(\bar{\mathbb{U}}, \bar{\mathbb{X}})$ *mediante los dos métodos expuestos para tres casos con diferentes conjuntos de restricciones en cada uno de ellos:*

$$\textit{Caso a:} \quad \mathbb{U}_1 \equiv \left\{ \begin{array}{c} -2 \leq u_1 \leq 1 \\ -1 \leq u_2 \leq 2 \end{array} \right\} \mathbb{U}_2 \equiv \left\{ \begin{array}{c} 1 \leq u_1 \leq 3 \\ -2 \leq u_2 \leq 3 \end{array} \right\}$$

$$\textit{Caso b:} \quad \mathbb{U}_1 \equiv \left\{ \begin{array}{c} -2 \leq u_1 \leq 4 \\ -1 \leq u_2 \leq 2 \end{array} \right\} \mathbb{U}_2 \equiv \left\{ \begin{array}{c} 1 \leq u_1 \leq 3 \\ -2 \leq u_2 \leq 3 \end{array} \right\}$$

$$\textit{Caso c:} \quad \mathbb{U}_1 \equiv \left\{ \begin{array}{c} -2 \leq u_1 \leq 1 \\ -0{,}5 \leq u_2 \leq 0{,}5 \end{array} \right\} \mathbb{U}_2 \equiv \left\{ \begin{array}{c} 2 \leq u_1 \leq 3 \\ -2 \leq u_2 \leq 3 \end{array} \right\}$$

| Caso | Alg. | # regs. | # LPs | CPU(s) |
|---|---|---|---|---|
| a | 1 | 28 | 3646 | 2.695 |
| | 2 | 3 | 300 | 0.764 |
| | 3 | 3 | 8157 | 109.334 |
| b | 1 | 115 | 53245 | 34.849 |
| | 2 | 4 | 558 | 1.072 |
| | 3 | 4 | 77702 | 53.872 |
| c | 1 | 1 | 1029 | 1.035 |
| | 2 | 1 | 741 | 1.246 |
| | 3 | 1 | - | - |

***Tabla 6.1:*** Comparación de métodos de cálculo de $\mathcal{O}_\infty^{\mathcal{K}_f}(\bar{\mathbb{U}}, \bar{\mathbb{X}})$.

*donde el caso a corresponde a las restricciones utilizadas en el ejemplo 6.1 y representadas en la figura 6.1 y las restricciones para los casos b y c están representados en las figuras 6.4 y 6.5 respectivamente. Para cada uno de los tres*

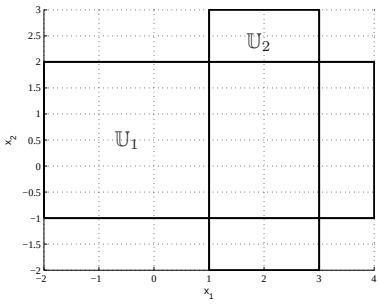


***Figura 6.4:*** Caso b.

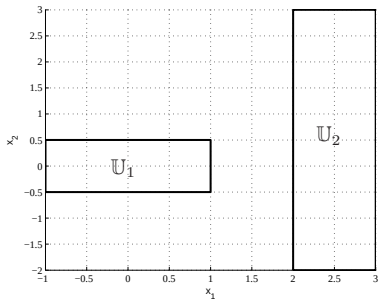


***Figura 6.5:*** Caso c.

*casos se comparan, con ayuda de la* MPT toolbox*, diferentes implementaciones en función de el número de regiones convexas cuya unión forma la región final, el número de problemas de programación lineal que es necesario resolver y el tiempo de cómputo de CPU en segundos en un Pentium D a 2.66 GHz y con 1 GB de memoria RAM. Los tres algoritmos comparados son:*

1. $\mathbb{K}_\infty(\bar{\mathbb{U}}, \Omega^{\mathcal{K}_f}, \Omega^{\mathcal{K}_f})$ *mediante el algoritmo 6.1.*
2. *Algoritmo 6.2 basado en* $\mathcal{O}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}})$.
3. *Algoritmo 6.1 y unión de regiones de [GTM08].*

*Los resultados se muestran en la tabla 6.1. Para los casos* a *y* b *puede comprobarse que el algoritmo 6.2 es mucho más eficiente que el algoritmo 6.1 tanto en términos de número de LPs resueltos como en tiempo de CPU. Esto es cierto considerando únicamente el algoritmo 6.1, pero la diferencia es aún mayor si además se aplica posteriormente el algoritmo de unión de regiones.*

*El caso* c *es especial en el sentido de que el conjunto de restricciones* $\mathbb{U}_2$ *no es de utilidad para aumentar* $\mathcal{O}_\infty^{\mathcal{K}_f}(\bar{\mathbb{U}}, \bar{\mathbb{X}})$*. Es decir, no existe ningún estado desde el cual el sistema en bucle cerrado evoluciones hacia* $\mathcal{O}_\infty^{\mathcal{K}_f}(\mathbb{U}_1, \bar{\mathbb{X}})$ *con una secuencia de control* $u_k = \mathcal{K}_f(x)$ *en el interior de* $\mathbb{U}_2$ *y la región* $\mathcal{O}_\infty^{\mathcal{K}_f}(\bar{\mathbb{U}}, \bar{\mathbb{X}})$ *sería exactamente la misma si únicamente se considerara* $\mathbb{U}_1$*. Esto significa que un subconjunto grande de* $\mathcal{O}_\infty^{\mathcal{K}_f}(\hat{\mathbb{U}}, \hat{\mathbb{X}})$ *no está incluido en* $\mathcal{O}_\infty^{\mathcal{K}_f}(\bar{\mathbb{U}}, \bar{\mathbb{X}})$*, haciendo menos adecuada la estrategia del algoritmo 6.2.*
*Desde el punto de vista del algoritmo 6.1, muchos de los conjuntos usados para calcular cada* $\mathbb{K}_{i+1}$*,* $\mathcal{Q}^{\mathcal{K}_f}(\mathbb{U}_2, \mathbb{K}_{ik}(\bar{\mathbb{U}}, \bar{\mathbb{X}}^{\mathcal{K}_f}, \bar{\mathbb{X}}^{\mathcal{K}_f}))$*, están vacíos, con lo que estos conjuntos están definidos por un número de poliedros mucho menor al considerado como peor caso.*
*Aún así, este algoritmo sigue teniendo un coste similar al del algoritmo 6.2, con un número mayor de LPs resueltos aunque un tiempo de CPU ligeramente inferior.* ■

## 6.4. Estabilidad asintótica

Al haberse demostrado las condiciones de estabilidad **C1-C5**, la estabilidad del control predictivo con restricciones no convexas en el sentido de Lyapunov queda demostrada con la elección adecuada de $\mathbb{X}_f$ por el teorema 6.1. No obstante, para que el sistema sea asintóticamente estable queda por probar la condición enunciada en el punto 3 de dicho teorema, es decir la continuidad de la función de coste en algún entorno del origen. Para ello, se formula el siguiente teorema.

**Teorema 6.5.** *El óptimo de la función de coste* $V_N^{OPT}(x)$ *para el problema* (6.1) *es continuo en un entorno del origen si éste no se encuentra en la frontera de ninguno de los conjuntos* $T_i$ *cuya unión forma el conjunto de restricciones no convexas.* □

**Prueba.** Tal y como se muestra en (3.5), el óptimo de la función de coste puede escribirse como

$$V_N^{OPT}(x) := \min \left\{ V_{iN}^{OPT}(x) \right\}$$

donde:

$$V_{iN}^{OPT}(x) := \min \tfrac{1}{2}\mathbf{u}^T H \mathbf{u} + \mathbf{u}^T F x,$$

sujeto a:
$\mathbf{u} \in T_i$

Puede demostrarse que las funciones $V_{iN}^{OPT}(x)$, que equivalen a la función de coste óptimas para un problema de control predictivo con restricciones convexas, son continuas [GS05].

Tomando los $\gamma_0$ conjuntos $T_i$ que contienen el origen e intersectándolos, como el origen no pertenece a la frontera de ningún $T_i$, esta intersección resulta

en un conjunto convexo que contiene el origen. En este conjunto, el mínimo puede descomponerse de la siguiente forma:

$$\begin{array}{ll} V_N^{OPT}(x) = \min \left\{V_{iN}^{OPT}(x)\right\} & \{i = 1 \dots \gamma_0\} \\ V_N^{OPT}(x) = \min \left\{V_{1N}^{OPT}(x), \min \left\{V_{iN}^{OPT}(x)\right\}\right\} & \{i = 2 \dots \gamma_0\} \\ \vdots & \end{array}$$

Por otro lado, dadas dos funciones cualesquiera, su mínimo puede escribirse como:

$$\min \{V_1(x), V_2(x)\} = \frac{1}{2}\left(V_1 + V_2 - |V_1 - V_2|\right)$$

Si $V_1(x)$ y $V_2(x)$ son continuas, teniendo en cuenta que la función *valor absoluto* también lo es, $\min \{V_1(x), V_2(x)\}$ es continua por ser una composición de funciones continuas definidas en el mismo espacio convexo.

Aplicando recursivamente la propiedad de continuidad del mínimo de dos funciones, queda probada la continuidad de $V_N^{OPT}(x)$ en un entorno del origen. ■

## 6.5. Cálculo del conjunto de estados iniciales factibles

El conjunto $\mathbb{S}_N$ puede calcularse, por aplicación del teorema 2.3, como:

$$\mathbb{S}_N = \mathcal{S}_N(\mathbb{X}, \mathbb{X}_f)$$

Para ello, sería necesario definir algoritmos sobre conjuntos de restricciones no convexos análogos a los descritos para $\mathbb{X}_f$. Esto, por causa de la explosión combinatoria, supondría un coste computacional elevado.

Una posible alternativa consiste en aprovechar que el controlador predictivo planteado se ha resuelto de manera explícita. Para ello, si se cumplen las condiciones **C1** − **C5** y la solución a (6.1) viene dada por

$$\mathbf{u}^{opt}(x) = G_i x + h_i, \quad \text{para } x \in X_i, \quad i = 0, \dots, N_r$$

donde $X_i$ es una partición del espacio de estados de la forma:

$$X_i := \left\{ x \in \mathbb{R}^n | \begin{array}{c} x^T M_{ij} x + N_{ij} x + C_{ij} \leq 0 \quad j = 1 \dots (\gamma - 1) \\ L_i x + W_i \leq 0 \end{array} \right\}$$

se tiene

$$\mathbb{S}_N = \bigcup_i^{N_r} X_i$$

Teniendo en cuenta además, que las curvas cuadráticas que definen regiones siempre lo hacen dividiendo una región lineal, pueden obviarse para simplificar

la unión de regiones anteriores. De esta forma, $\mathbb{S}_N$ se calculará como unión de poliedros, y por tanto podrán aplicarse los algoritmos utilizados en secciones anteriores.

A continuación se introduce un procedimiento que puede utilizarse en caso de querer obtener el conjunto de estados iniciales factibles sin necesidad de obtener previamente la solución explícita. Para ello, se parte del problema (6.2)

$$\begin{aligned}\mathcal{P}_N(x) : & V_N^{OPT}(x) := \min\ V_N(x,u)\\ & \text{sujeto a:}\\ & u_k \in \bar{\mathbb{U}} \text{ para } k = 0,\dots,N-1,\\ & x_k \in \bar{\mathbb{X}} \text{ para } k = 0,\dots,N,\\ & x_N \in \bar{\mathbb{X}}_f \subset \mathbb{X}\end{aligned}$$

donde:

$$\bar{\mathbb{U}} = \bigcup_{i=0}^{\gamma_u} \mathbb{U}_i \quad \bar{\mathbb{X}} = \bigcup_{i=0}^{\gamma_x} \mathbb{X}_i \quad \bar{\mathbb{X}}_f = \bigcup_{i=0}^{\gamma_f} \mathbb{X}_{f,i}$$

Dicho problema, como se ha visto, es equivalente a un problema en que el óptimo es el mínimo de los diferentes subproblemas:

$$\mathcal{P}_{N,M}(x): \quad V_{N,M}^{OPT}(x) := \min\ \left\{V_{iN}^{OPT}(x)\right\}$$

donde:

$$\begin{aligned}& V_{iN}^{OPT}(x) := \min\ \tfrac{1}{2}\mathbf{u}^T H \mathbf{u} + \mathbf{u}^T F x,\\ & \text{sujeto a:}\\ & (\mathbf{u}, x) \in T_i\end{aligned}$$

Como se ha visto, cada uno de estos subproblemas es equivalente a un problema de control predictivo con restricciones convexas, y por tanto tendrá una serie de estados iniciales para los que se puede garantizar factiblidad y estabilidad, $\mathbb{S}_N^i$. Dicho conjunto, puede calcularse como la unión de todas las regiones de la solución explícita o, por aplicación del teorema 2.3.

Por último, dado que $\bar{\mathbb{U}} = \bigcup \mathbb{U}_i$, $\bar{\mathbb{X}} = \bigcup \mathbb{X}_j$ y $\bar{\mathbb{X}}_f = \bigcup \mathbb{X}_{f,l}$ los conjuntos iniciales factibles para el problema (6.2) pueden obtenerse como:

$$\mathbb{S}_N = \bigcup_i \mathbb{S}_N^i$$

## 6.6. Conclusiones

En este capítulo, se han analizado las condiciones de estabilidad *a priori* para el control predictivo de sistemas lineales con restricciones poliédricas no convexas. Se ha demostrado que, para este tipo de sistemas existen dos diferencias fundamentales con respecto a los sistemas con restricciones poliédricas:

- El conjunto de restricciones terminal óptimo, máximo conjunto invariante en bucle cerrado, es un poliedro no convexo.
- El índice de coste es continuo en un entorno del origen.

La continuidad del índice de coste permite asegurar que, si se cumplen todas las condiciones de estabilidad, el sistema en bucle cerrado no es únicamente estable en el sentido de Lyapunov, sino asintóticamente estable.

En cuanto al máximo conjunto invariante, se plantean dos algoritmos diferentes para su cálculo. Se muestra como el algoritmo más sencillo, basado en una extensión del algoritmo usado para restricciones convexas y unas proposiciones para el cálculo del conjunto a un paso para poliedros no convexos, es poco eficiente en cuanto al número de regiones que se obtienen. En su lugar, se plantea un algoritmo alternativo, basado en el máximo conjunto invariante para las envolventes convexas de las restricciones, y se muestra para diferentes ejemplos que es en general más eficiente.

Capítulo 7

# Aplicaciones

## 7.1. Introducción

En este capítulo se introducen algunos problemas de la ingeniería de control para los que son aplicables las técnicas de control predictivo con restricciones poliédricas no convexas desarrolladas. Las aplicaciones en los diferentes ámbitos se ilustran con ejemplos apropiados.
En la siguiente sección, se proponen aquellas aplicaciones que, por su propia naturaleza, presentan restricciones de la forma propuesta en capítulos anteriores. En la tercera sección se presenta un problema más general con restricciones no lineales no convexas, desarrollando una solución subóptima mediante la sustitución de dichas restricciones por uniones no convexas de poliedros. Además, se muestra como el problema con restricciones no lineales aparece en técnicas de control no lineal muy extendidas, como el control por cancelación de la no linealidad de sistemas Hammerstein y Wiener y el control mediante linealización por realimentación.

## 7.2. Aplicaciones con restricciones poliédricas no convexas puras

Las restricciones poliédricas no convexas aparecen de forma natural en problemas como el de la evitación de obstáculos, inherentemente no convexo. En [Kur05] puede comprobarse la importancia de este problema, que además aparece de manera natural en multitud de problemas de la ingeniería de control, como la planificación de trayectorias de robots [Kha86] o la planificación de tráfico de vehículos voladores, [KY00].

Se considera el sistema lineal invariante de tiempo discreto

$$x_{k+1} = Ax_k + Bu_k$$

con $x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$ y sujeto a las restricciones:

$$x_k \in \mathbb{X}$$
$$u_k \in \mathbb{U}$$
$$x_k \notin \mathbb{Z}$$
$$u_k \notin \mathbb{V}$$

Dichas restricciones, pueden reescribirse de la forma:

$$x_k \in \mathbb{X}_\mathbb{Z}$$
$$u_k \in \mathbb{U}_\mathbb{V}$$

donde:

$$\mathbb{X}_\mathbb{Z} = \mathbb{X} \backslash \mathbb{Z}$$
$$\mathbb{U}_\mathbb{V} = \mathbb{U} \backslash \mathbb{V}$$

Resulta evidente que, aún si todos los conjuntos $\mathbb{X}$, $\mathbb{U}$, $\mathbb{Z}$ y $\mathbb{V}$ son convexos, $\mathbb{X}_\mathbb{Z}$ y $\mathbb{U}_\mathbb{V}$ no tienen por qué serlo, y por tanto vendrán definidos como la unión de varios poliedros convexos

$$\mathbb{X}_\mathbb{Z} = \bigcup_i \mathbb{X}_{\mathbb{Z}i}$$
$$\mathbb{U}_\mathbb{V} = \bigcup_j \mathbb{U}_{\mathbb{V}j}$$

que pueden ser empleados como restricciones de un controlador predictivo como los diseñados en capítulos anteriores.

Para sistemas discretos, una forma de abordar el problema es la mostrada en [RM05], que calcula los conjuntos alcanzables, es decir, el conjunto de estados desde los que se puede alcanzar un conjunto de destino. También existen enfoques de control predictivo para esta clase de problemas, como el mostrado en [RM07], pero se basan en la resolución de un problema de programación cuadrática entera mixta en línea, lo que puede ser demasiado costoso computacionalmente.

### 7.2.1. Ejemplo de aplicación: Robot cartesiano

Se considera el control de posición de un robot cartesiano con dos motores de corriente continua independientes como manipuladores. Para mayor simplicidad, se supone un control de par sobre los motores, de forma que se consideran como acciones de control las corrientes en cada uno de ellos. Por tanto, los estados del sistema son $x_1 = \omega_1$, $x_2 = \omega_2$, $x_3 = \theta_1$ y $x_4 = \theta_2$ y las entradas $u_1 = i_1$

y $u_2 = i_2$.
El modelo continuo del sistema es

$$\begin{cases} \dot{x} = \begin{bmatrix} -\frac{B_1}{J_1} & 0 & 0 & 0 \\ 0 & -\frac{B_2}{J_2} & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} x + \begin{bmatrix} \frac{K_1}{J_1} & 0 \\ 0 & \frac{K_2}{J_2} \\ 0 & 0 \\ 0 & 0 \end{bmatrix} u \\ y = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} x \end{cases}$$

donde $B_i$,$J_i$ y $K_i$ son, respectivamente, el coeficiente de fricción viscosa, la inercia total y la constante de par de los motores, que toman los siguientes valores: $B_1 = B_2 = 0{,}02Nms$, $J_1 = 0{,}06kgm^2$, $J_2 = 0{,}04kgm^2$, $K_1 = 2Nm/A$ y $K_2 = 1{,}4Nm/A$. El sistema se discretiza con un período $T_s = 0{,}2$.

El proceso está además sujeto a restricciones convexas sobre las corrientes, $|i_i| \leq 3A$, y velocidades, $|\omega_i| \leq 200rad/s$ pero no convexas sobre la posición, que debe mantenerse en el interior de una región $\mathbb{X}$ pero fuera de otra región correspondiente a un obstáculo que es necesario esquivar:

$$(x_3, x_4) \in \mathbb{X} := \left\{ \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x_3 \\ x_4 \end{bmatrix} \leq \begin{bmatrix} 6 \\ 6 \\ 1 \\ 1 \end{bmatrix} \right\}$$

$$(x_3, x_4) \notin \mathbb{Z} := \left\{ \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x_3 \\ x_4 \end{bmatrix} \leq \begin{bmatrix} 5 \\ 6 \\ 1 \\ -1 \end{bmatrix} \right\}$$

Como se ha visto, las restricciones anteriores pueden escribirse como un conjunto poliédrico no convexo, en este caso de dos regiones, $(x_3, x_4) \in \mathbb{X}_\mathbb{Z} = \mathbb{X}_1 \cup \mathbb{X}_2$ donde:

$$\mathbb{X}_1 := \left\{ \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x_3 \\ x_4 \end{bmatrix} \leq \begin{bmatrix} 6 \\ 1 \\ 1 \\ 1 \end{bmatrix} \right\} \quad \mathbb{X}_2 := \left\{ \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} x_3 \\ x_4 \end{bmatrix} \leq \begin{bmatrix} 6 \\ 6 \\ -5 \\ 1 \end{bmatrix} \right\}$$

Los parámetros de diseño para el controlador MPC son $N = 2$, $Q = C^T C$ y $R = \begin{bmatrix} 1 & 0 \\ 0 & 10 \end{bmatrix}$. Se calculan también el controlador y la región terminal que garantizan la estabilidad del sistema, estando ésta última formada por dos poliedros. Para $N = 2$, hay por tanto 4 conjuntos de restricciones $T_i$ diferentes. Calculando la envolvente convexa y resolviendo el mpQP correspondiente, se obtiene una solución explícita de 148 regiones.
Tras la clasificación de regiones, división de las no factibles y unión de las que tienen la misma solución, se obtiene una partición de 387 regiones.
A continuación, se aplica el algoritmo de eliminación de soluciones no óptimas, lo que permite eliminar 339 soluciones mediante la resolución de 634 problemas

de suma de cuadrados.
Por último, para obtener la partición final, se realiza una vez más la unión de regiones con una misma solución. Tras ello, se tienen 246 regiones, una de ellas con 4 soluciones, otra con 3, 68 con dos y el resto con sólo una.

La figura 7.1 muestra la evolución de la posición del robot cartesiano, que claramente satisface las restricciones. No obstante, al ser el proceso real continuo, debe prestarse atención para seleccionar un período de muestreo suficientemente bajo, puesto que el controlador solo garantiza el cumplimiento de las restricciones en los instantes de muestreo.

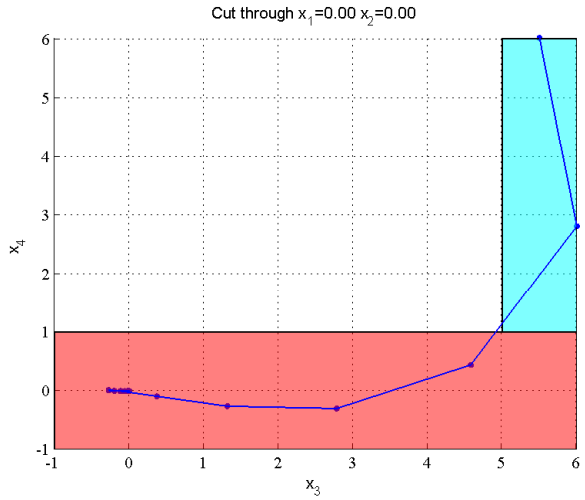


***Figura 7.1:*** Sección de la partición final poliédrica y trayectoria.

En cuanto al algoritmo en línea, se obtiene un árbol binario y se compara el coste con el de obtener un árbol para cada uno de los conjuntos $T_i$ y comparar los índices de coste. El árbol binario de la solución obtenida tiene una profundidad $D = 12$, frente a las de los otros cuatro, $D_1 = 9$, $D_2 = 7$, $D_3 = 8$ y $D_4 = 6$. Siguiendo el análisis del capítulo 6, el número total de operaciones elementales es de $C_1 = 465$ y $C_2 = 303$, lo que supone una reducción del coste computacional de un 34.8 %. En cuanto a las necesidades de memoria, para el algoritmo básico se requiere almacenar 7440 números reales y 676 punteros, mientras que para el árbol de la solución completa son almacenados 3285 números reales y 901 punteros. Suponiendo que el controlador se implementa en un microcontrolador de 16 bits en el que los números reales requieren el doble de memoria que los punteros, la reducción de memoria a utilizar es de un 51.97 %.

## 7.3. Aplicaciones con restricciones no lineales

Se desea formular un controlador predictivo con modelos lineales y restricciones no lineales con el problema de optimización asociado (7.1)

$$\begin{aligned}\mathcal{P}_{N,M}(x) : V_{N,M}^{OPT}(x) := \min\ & \frac{1}{2}\mathbf{u}^T H\mathbf{u} + \mathbf{u}^T Fx \\ & \text{sujeto a:} \\ & x_{k+1} = Ax_k + Bu_k \text{ para } k = 0, \ldots, N-1, \\ & x_0 = x, \\ & f_i(\mathbf{u}, x) \leq 0 \text{ para } i = 1, \ldots, N_f\end{aligned} \tag{7.1}$$

donde se han incluido $N_f$ desigualdades que pueden ser funciones no lineales cualesquiera dependientes del estado $x$, conocido en el momento de calcular la acción de control y de las $M$ acciones de control futuras. Dichas desigualdades, de carácter muy general, pueden incluir restricciones añadidas con objeto de garantizar la estabilidad del esquema de control predictivo.
Si se desea resolver el problema anterior de manera exacta, es necesario recurrir a la programación no lineal, con la complejidad computacional en línea que ello conlleva.
Si dicho coste en línea no es aceptable, puede definirse, en su lugar un problema de la forma (3.3):

$$\begin{aligned}\mathcal{P}_{N,M}(x) : \quad & V_{N,M}^{OPT}(x) := \min\ \tfrac{1}{2}\mathbf{u}^T H\mathbf{u} + \mathbf{u}^T Fx, \\ & \text{sujeto a:} \\ & (\mathbf{u}, x) \in \bar{\mathbb{T}}\end{aligned}$$

donde:

$$\bar{\mathbb{T}} = \bigcup_{i=0}^{\gamma} T_i$$

$$T_i := \left\{ (\mathbf{u}, x) \in \mathbb{R}^{Mm+n} \,\middle|\, \begin{bmatrix}\Phi_i & \Lambda_i\end{bmatrix} \begin{bmatrix}\mathbf{u} \\ x\end{bmatrix} \leq \Delta_i \right\}$$

Como se ha visto en capítulos anteriores, dicho problema admite el cálculo de una solución explícita con un tiempo de coste en línea aceptable para problemas de tamaño no demasiado elevado.
La necesidad que se plantea en este punto es cómo definir un problema de la forma (3.3) que sea una buena aproximación para el problema (7.1). Dado que para ambos problemas el índice de coste es el mismo, la decisión se reduce a escoger el conjunto poliédrico no convexo $\mathbb{T}$.
Para la obtención de dicho conjunto, se pueden establecer dos grandes objetivos:

1. El sistema en bucle cerrado que se obtiene con las restricciones originales (7.1) y la acción de control resultante de resolver el problema (3.3) debe resultar *estable*.

2. El conjunto de estados iniciales estabilizables de este modo debe ser lo más grande posible.

El primer objetivo puede satisfacerse mediante el diseño de un problema estable con restricciones poliédricas no convexas, conforme se ha descrito en el capítulo 6 siempre y cuando las acciones de control que se obtengan para dicho problema sean también factibles en el problema real. Es decir, el conjunto poliédrico no convexo debe ser una aproximación interior del conjunto factible definido por las restricciones $f_i$:

$$(\mathbf{u}, x) \in \bar{\mathbb{T}} \Rightarrow f_i(\mathbf{u}, x) \leq 0 \qquad \forall i \tag{7.2}$$

En cuanto al segundo objetivo, el conjunto de estados estabilizables es mayor cuanto mayor sea el conjunto de restricciones $\mathbb{T}$, por tanto será conveniente que éste se aproxime lo más posible al espacio definido por las restricciones $f_i$.
Por último, es importante recordar al diseñar el conjunto $\mathbb{T}$ que la complejidad de la solución explícita del problema (7.1) depende enormemente del número de conjuntos convexos $T_i$, $\gamma$.

### 7.3.1. Aproximación de no linealidades mediante unión de poliedros

En este punto, se parte de una serie de restricciones no lineales $f_i \leq 0$ que se desea substituir por un poliedro no convexo lo más grande posible que satisfaga las restricciones originales. Para ello, el problema se abordará en dos fases, primero sustituir cada una de las funciones no lineales por una función afín a tramos y, una vez se disponga de todas ellas, formar los poliedros no convexos.
Definiremos una función afín a tramos como

$$F(x) = a_j x + b_j \quad \text{para } R_j x \leq r_j$$

Por tanto, nuestro primer problema consiste en buscar, para cada función $f_i(x)$, la función $F_i(x)$ que mejor la aproxime en el espacio estudiado $X$ cumpliendo:

$$F_i(x) < f_i(x) \quad \forall x \in X \tag{7.3}$$

Desafortunadamente, se trata de un problema abierto para el que no existe una solución definitiva para cualquier tipo de funciones. No obstante, para algunas tipologías concretas, sí existen soluciones propuestas en la bibliografía.
En particular, cuando la función es cóncava (lo que implica que el espacio definido por $f_i(x) \leq 0$ es convexo) una aproximación sencilla puede obtenerse simplemente evaluando la función en diferentes puntos y uniéndolos (obteniendo la envolvente convexa). El cumplimiento de la condición (7.3) queda garantizado por la propia concavidad de la función.
Cuando la función es convexa (y por tanto el espacio cóncavo), puede aproximarse mediante los subgradientes [BY09]:

$$F(x) = max\{f(x_0) + y_0^T(x - x_0), \ldots, f(x_k) + y_k^T(x - x_k)\}$$

donde $y_i$ es un subgradiente de la función en $x_i$. Un vector $y \in \mathbb{R}^n$ es un subgradiente de la función $f$ en un punto $x$ si:

$$f(z) \geq f(x) + y^T(z - x), \quad \forall z \in \mathbb{R}^n$$

Cuando la función es derivable, el subgradiente puede ser el gradiente.
Ninguna de las dos opciones anteriores (envolvente convexa o subgradientes) es aplicable directamente si la función no es cóncava ni convexa. No obstante, en estos casos, y siempre que la función pueda escribirse mediante suma de términos algebraicos, ésta puede separarse en términos convexos, cóncavos univariables y bilineares [SP99]. Posteriormente, puede encontrarse una función afín a tramos que acote cada uno de ellos. Este es el enfoque seguido por [BAG05].

Un caso de función no lineal particular que es interesante considerar es el de las *splines* (curvas polinomiales a tramos) puesto que existen multitud de métodos para aproximar mediante ellas otras funciones no lineales (con una precisión creciente con el número de puntos que se toman). Para este tipo de funciones, existen métodos eficientes que calculan su *enclosure*, que son curvas afines a tramos que la encierran [LP01].

Otra opción existente es la propuesta en [CAF03], que resulta interesante por su sencillez. En este trabajo los autores proponen definir previamente las regiones $(R_j, r_j)$ sobre las que se definirá la función $F(x)$ y garantizar, mediante la resolución de un problema de programación lineal, que ésta acote a $f(x)$ en un conjunto discreto de puntos. Aunque presenta el inconveniente de que no existen garantías fuera de dichos puntos, puede ser suficiente si la función no es demasiado compleja y se toman suficientes puntos.

Una vez se tienen las no linealidades aproximadas mediante alguno de los métodos citados anteriormente, el espacio de restricciones vendrá determinado por una serie de desigualdades lineales cuya intersección es un poliedro (provenientes de restricciones originalmente lineales o aproximaciones lineales a tramos de no linealidades convexas) y un determinado número de funciones lineales a tramos provenientes de la aproximación de no linealidades no convexas:

$$\begin{cases} R_c x \leq r_c \\ F_i(x) \geq 0 \quad i = 1 \dots n_t \end{cases}$$

donde

$$F_i(x) = max\{a_{ij}x + b_{ij}\} \qquad j = 1 \dots n_i$$

Con objeto de obtener el espacio de restricciones como un poliedro no convexo, se pueden formar todos los poliedros que aparecen al tomar todas las combinaciones posibles de desigualdades lineales que definen cada una de las curvas lineales a tramos. De esta forma, se tendrán $\prod_i^{n_t} n_i$ posibles poliedros cuya unión forma el espacio de restricciones. No obstante, algunos de ellos pueden ser no factibles y/o estar definidos por restricciones redundantes.
Dichos poliedros no factibles y restricciones redundantes pueden ser eliminados mediante resolución de problemas de programación lineal siguiendo el procedimiento descrito para la intersección de particiones poliédricas en la sección 4.2.

Una vez en este punto, con una serie de poliedros no vacíos y sin restricciones redundantes (aunque con posibles solapamientos) es posible aplicar uno de los algoritmos de [GTM08] para obtener un número mínimo, de manera análoga a como se hizo para la minimización del número de regiones en una partición en el algoritmo 4.8, obteniendo de esta forma una representación de las restricciones como la del problema (3.3) con un número mínimo de conjuntos convexos $\gamma$.

**Ejemplo 7.1.** *Se tiene un espacio de restricciones definido por restricciones poliédricas y dos funciones afines a tramos no convexas, provenientes de la aproximación de dos restricciones no lineales:*

$$\begin{cases} \begin{bmatrix} 0 & -1 \\ -1 & 0 \\ 0 & 1 \\ 1 & 0 \end{bmatrix} x \leq \begin{bmatrix} 0 \\ 0 \\ 2 \\ 2 \end{bmatrix} \\ max\{-x_2+1, x_1-1, x_1-x_2+0{,}5\} \geq 0 \\ max\{-x_1+1, x_2-0{,}5\} \geq 0 \end{cases}$$

*representado en la figura 7.2.*
*Si se obtienen por enumeración de los tramos de las funciones no lineales todos*

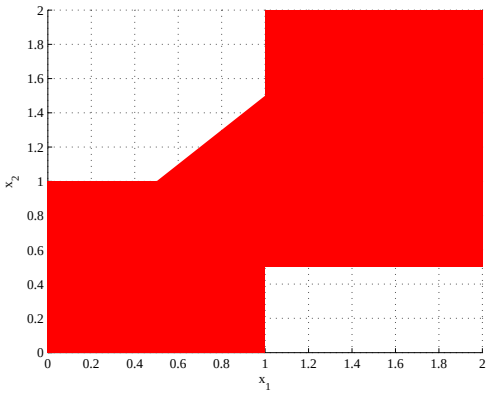


***Figura 7.2:*** Espacio de restricciones no convexo.

*los poliedros posibles, hay un máximo de* 6 *diferentes (puesto que las funciones son de 3 y 2 tramos respectivamente). No obstante, uno de ellos no es factible, por lo que quedan* 5 *poliedros, representados en la figura 7.3 (tras eliminar para cada uno de ellos las restricciones redundantes).*
*Con los poliedros formados, es posible aplicar uno de los algoritmos de [GTM08] para obtener el conjunto de restricciones formado por la unión no convexa de* 3 *poliedros mostrado en la figura 7.4.* ■

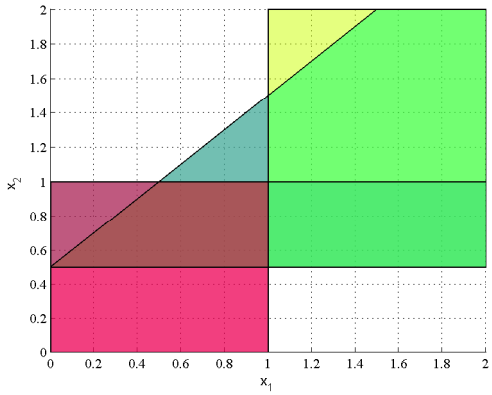


***Figura 7.3:*** Espacio de restricciones con poliedros solapados.

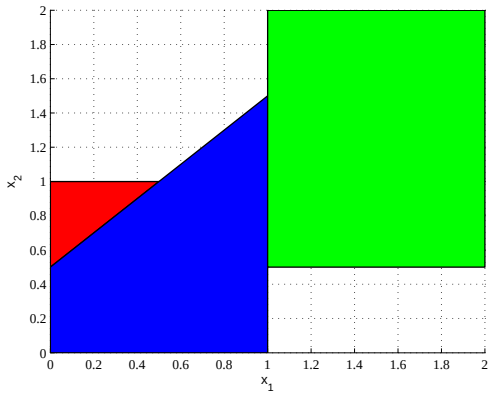


***Figura 7.4:*** Espacio de restricciones con número mínimo de poliedros.

### 7.3.2. Control Predictivo de sistemas Hammerstein y Wiener

Existen muchos procesos no lineales cuya dinámica puede ser aproximada de manera satisfactoria mediante sistemas lineales puesto que el comportamiento no lineal se encuentra concentrado en la ganancia estática del proceso. Este tipo de sistemas pueden ser modelados mediante estructuras de tipo Hammerstein y/o Wiener.
Un sistema Hammerstein está formado por una linealidad estática seguida de un sistema dinámico lineal (representado, por ejemplo, en espacio de estados):

$$\begin{aligned} v_k &= h(u_k) \\ x_{k+1} &= Ax_k + Bv_k \\ y_k &= Cx_k + Dv_k \end{aligned}$$

donde $u_k \in \mathbb{R}^m$, $y_k \in \mathbb{R}^p$, $x_k \in \mathbb{R}^n$, $v_k \in \mathbb{R}^m$ son las entradas ficticias intermedias del sistema lineal y $h : \mathbb{R}^m \to \mathbb{R}^m$ es la función no lineal estática.
Análogamente, un sistema Wiener está formado por un sistema dinámico seguido de una función no lineal estática:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k \\ z_k &= Cx_k + Du_k \\ y_k &= w(z_k) \end{aligned}$$

siendo en este caso $z_k \in \mathbb{R}^p$ las salidas ficticias intermedias y $w : \mathbb{R}^p \to \mathbb{R}^p$ es la función no lineal estática.
Ambas estructuras pueden combinarse de manera simultánea para formar un sistema Hammerstein-Wiener (figura 7.5), capaz de representar sistemas no lineales más complejos:

$$\begin{aligned} v_k &= h(u_k) \\ x_{k+1} &= Ax_k + Bv_k \\ z_k &= Cx_k + Dk_k \\ y_k &= w(z_k) \end{aligned}$$

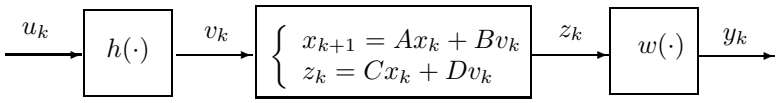


***Figura 7.5:*** Sistema Hammerstein-Wiener.

Si se desea plantear un control predictivo de un sistema de este tipo, la opción más directa consiste en desarrollar un modelo de predicción y aplicar técnicas de programación no lineal para resolver el problema de optimización que aparece [PLS97], [MPJMM09].
No obstante, existen otras propuestas que tratan de explotar la estructura particular del sistema no lineal. Entre estas, muchas de las metodologías más extendidas se basan en la cancelación de las no linealidades estáticas [FPM97], [PLS98], [NPR99], [ABB+00], [GJB04] . La idea fundamental consiste en obtener inversas de las funciones $h$ y $w$ (bien porque éstas son conocidas e invertibles, o bien identificando directamente la función inversa).
De esta forma, a partir de la medida de las salidas del sistema $y_k$ pueden obtenerse las salidas ficticias $z_k$:

$$z_k = w^{-1}(y_k)$$

Con ellas, es posible diseñar un control predictivo para la parte lineal del sistema, que proporcionará las acciones de control ficticias futuras $v_k$, a partir de las cuales pueden calcularse las acciones de control reales a aplicar:

$$u_k = h^{-1}(v_k)$$

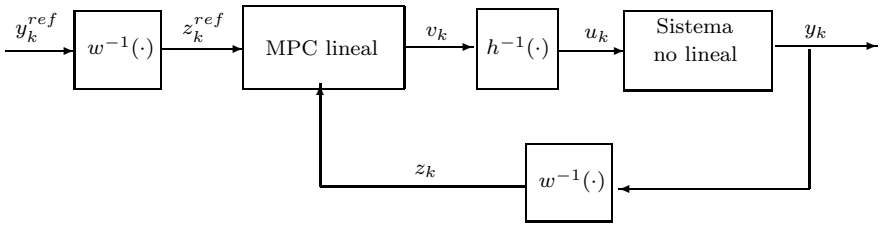


***Figura 7.6:*** Control por cancelación de no linealidades de un sistema Hammerstein-Wiener.

La ventaja obvia de las estrategias de cancelación de las no linealidades es su sencillez, puesto que prácticamente permite tratar el sistema no lineal como uno lineal, con todas las herramientas desarrolladas para éstos, lo que permite su aplicación a procesos rápidos.
No obstante, existen una serie de dificultades que en la práctica dificultan la estrategia de cancelación:

- Representación invertible de la no linealidad. Si las funciones $h$ y $w$ no son invertibles, es imposible construir la estructura de control propuesta.
- Aunque las funciones sean invertibles, en un caso real las no linealidades nunca se cancelarán de una manera totalmente exacta.
- Identificación. En general, la identificación es más compleja que en modelos lineales, debido al mayor número de parámetros que es necesario identificar por la presencia de las no linealidades estáticas.
- No optimalidad del índice minimizado: Si se realizan las transformaciones propuestas, en el índice no se incluyen las acciones de control y salidas reales, sino las variables ficticias introducidas.
- Tratamiento de restricciones: Las restricciones existentes en el sistema vienen dadas sobre las salidas y acciones de control reales. No obstante, la estructura de control propuesta sólo es capaz de manejar de manera explícita restricciones sobre las variables ficticias.

Aunque todos los problemas enumerados son tratados en mayor o menor medida en las diferentes propuestas bibliográficas, cuando se plantea la introducción de un control predictivo para el diseño del controlador de la parte lineal del sistema entre todas las dificultades cobra una especial importancia la relativa al tratamiento de restricciones realizado. Esto es así debido a que uno de los puntos fuertes que justifica la utilización de controladores predictivos es precisamente la inclusión de las restricciones desde un principio en el propio diseño del controlador. Por esta razón, a continuación se describe brevemente cómo abordan esta cuestión las referencias bibliográficas consultadas.

Es importante notar que, si se desea evitar la programación no lineal, el tratamiento de las restricciones en este tipo de problemas es considerablemente

dificultoso, por lo que no existe ninguna referencia definitiva. De hecho algunas propuestas, como [Jur06] o [PLS98], directamente las ignoran en la formulación. Otras, como [GJB04], sólo consideran sistemas de tipo Wiener y restricciones sobre las acciones de control, pero no sobre las salidas, que son las que presentarían dificultades. En [NPR99] también se trabaja con sistemas Wiener y, aunque sí se consideran restricciones a la salida, éstas se incluyen en la formulación del problema mediante un procedimiento *adhoc* que no ofrece garantías de satisfacción.
Existen también varias referencias que permiten trabajar con restricciones de saturación sobre las acciones de control y/o salidas transformándolas a su vez en saturaciones sobre las variables ficticias. Este tipo de procedimientos es o bien subóptimo, puesto que reduce el espacio de acciones de control a disposición del controlador, o bien de limitada aplicación, puesto que sólo es válido para representaciones muy concretas de las funciones no lineales. Entre las referencias con este enfoque, destacan [FPM97] y [ABB+00] para sistemas Hammerstein y [CAF03] para sistemas Wiener.
Por último existen algunos otros enfoques, como el de [BdBV01] que no se basa en una cancelación de la no linealidad, pero sí hace uso de la inversa de ésta. Esta propuesta obtiene una representación politópica de las no linealidades y resuelve el problema como si se tratara de un sistema incierto (control predictivo robusto). No obstante, requiere la resolución en línea de SDPs, lo que dificulta su aplicabilidad a procesos rápidos.

A la vista de la problemática existente para el tratamiento de las restricciones con cancelación de la no linealidad en estos sistemas, se va a plantear el problema desde el enfoque del control predictivo con restricciones poliédricas no convexas.
Se considera en primer lugar un sistema Hammerstein-Wiener como el de la figura 7.5 para el que se tienen unas restricciones lineales sobre la acción de control

$$Ru_k \leq r \tag{7.4}$$

Conocida la no linealidad estática $v_k = h(u_k)$, y suponiendo que puede invertirse, la acción de control real puede calcularse a partir de la ficticia:

$$u_k = \begin{bmatrix} h_1^{-1}(v_k) \\ \vdots \\ h_m^{-1}(v_k) \end{bmatrix} \tag{7.5}$$

Sustituyendo (7.5) en (7.4), se tiene:

$$\begin{aligned} R_{11}h_1^{-1}(v_k) + \ldots + R_{1m}h_m^{-1}(v_k) &\leq r_1 \\ &\vdots \\ R_{l1}h_1^{-1}(v_k) + \ldots + R_{lm}h_m^{-1}(v_k) &\leq r_l \end{aligned} \tag{7.6}$$

donde $R_{ij}$ representa el elemento $(i, j)$ de la matriz $R$.
En el esquema de control predictivo se desea que toda la secuencia de control satisfaga las restricciones (7.4) por lo que las desigualdades (7.6) deben cumplirse para $k = 0 \ldots M - 1$.

Por otra parte, si el sistema tiene restricciones sobre la salida (7.7), es necesario hacer uso del modelo de predicción para la parte lineal.

$$R_y y_k \leq r_y \tag{7.7}$$

Dicho modelo, siguiendo la terminología del capítulo 2, puede escribirse como

$$\mathbf{z} = \Omega x + \Gamma \mathbf{v} \tag{7.8}$$

donde $\mathbf{z} = [z_1 \dots z_{N-1}]^T$ y $\mathbf{v} = [v_0 \dots v_{M-1}]^T$ representan respectivamente las secuencias de salidas y acciones de control ficticias futuras.
Sustituyendo la no linealidad a la entrada en la ecuación (7.8), se tiene

$$\mathbf{z} = \Omega x + \Gamma [v_0 \dots v_{M-1}]^T$$

Para un sistema MIMO con $m$ entradas, la acción de control ficticia en cada instante puede escribirse como $v_k = [v_{k1} \dots v_{km}]^T$:

$$\begin{aligned} \mathbf{z} &= \Omega x + \Gamma [v_{01} \dots v_{0m} \dots v_{(M-1)1} \dots v_{(M-1)m}]^T \\ \mathbf{z} &= \Omega x + \Gamma [h_1(u_0) \dots h_m(u_0) \dots h_1(u_{M-1}) \dots h_m(u_{M-1})]^T \end{aligned} \tag{7.9}$$

Por otra parte, conocida la no linealidad a la salida, $y_k = w(z_k)$, se tiene:

$$\mathbf{y} = [y_1 \dots y_N]^T = [w(z_1) \dots w(z_N)]^T$$

por lo que (7.7) puede escribirse como

$$R_y [w(z_1) \dots w(z_N)]^T \leq r_y \tag{7.10}$$

Substituyendo (7.9) en (7.10), se tiene:

$$\begin{aligned} &R_y \left[ w \left( \Omega_1 x + \Gamma_1 [h_1(u_0) \dots h_m(u_0) \dots h_1(u_{M-1}) \dots h_m(u_{M-1})]^T \right) \dots \right. \\ &\left. w \left( \Omega_N x + \Gamma_N [h_1(u_0) \dots h_m(u_0) \dots h_1(u_{M-1}) \dots h_m(u_{M-1})]^T \right) \right]^T \leq r_y \end{aligned} \tag{7.11}$$

donde $\Omega_i$ y $\Gamma_i$ representan la fila $i$ de las respectivas matrices.

Tanto las restricciones a la entrada (7.6) como las restricciones a la salida (7.11) pueden representarse mediante una serie de funciones no lineales genéricas de la forma

$$f_i(\mathbf{u}, x) \leq 0$$

Por tanto, es aplicable la aproximación mediante poliedros detallada en la sección 7.3.1 y el control predictivo mediante restricciones poliédricas no convexas.

### 7.3.3. Ejemplo de aplicación: Control de un motor Diesel sobrealimentado.

Los motores Diesel destinados a vehículos de pasajeros son procesos muy complejos que son continuamente modificados y desarrollados para la mejora de una serie de objetivos. Por un lado se encuentran los requerimientos del usuario del vehículo:

- Alta potencia.
- Bajo consumo.
- Elasticidad en la conducción.

Por otro lado, existen una serie de normativas que deben cumplirse. Entre éstas, destacan los límites sobre la máxima emisión a lo largo de dos tipos de ciclo de test (uno urbano y otro extra urbano) de diferentes tipos de contaminantes, fundamentalmente óxidos de nitrógeno($NO_x$) y humos.

### Proceso de renovación de la carga

Para la mejora de la potencia desarrollada y la conducción, la mayoría de motores diesel actuales utilizan un turbocompresor (fig. 7.7). Éste utiliza una porción de la energía de los gases de escape (producto de la combustión) para incrementar la cantidad de aire que se introduce en los cilindros. Esta mayor cantidad de aire permite quemar una mayor cantidad de combustible, consiguiendo mayores valores de potencia y par motor que un motor diesel atmosférico. Típicamente, un turbocompresor consiste en una turbina y un compresor acoplados por un eje común, normalmente dimensionados para obtener una respuesta rápida en la masa de aire cuando el conductor demanda aceleración a bajas velocidades. No obstante, un turbocompresor dimensionado de esta forma podría dañar el motor debido a las elevadas presiones que aparecerían en el colector de admisión a velocidades más altas. Una posible solución es utilizar una válvula *wastegate* que permite desviar parte de los gases de escape de forma que no circulen a través de la turbina a altas velocidades. Otra posibilidad, actualmente más utilizada, es usar una turbina de geometría variable ($TGV$). Ésta puede ser dimensionada para cada velocidad del motor directamente durante el funcionamiento, variando el área de flujo y el ángulo con el que los gases de escape se dirigen a los álabes de la turbina.

Por otro lado, para reducir la emisión de $NO_x$ se recircula una parte de los gases de escape de vuelta al colector de admisión. Es lo que se conoce como *recirculación de gases de escape* ($EGR$) y típicamente se consigue mediante una válvula que conecta los colectores de admisión y escape. Los gases recirculados, de alto calor específico, actúan como gases inertes, disminuyendo la temperatura de la combustión, y por tanto la velocidad de la reacción de formación de $NO_x$.

En cuanto a los humos, en particular la cantidad de partículas, ésta depende fundamentalmente del ratio aire-fuel ($AFR$) en la combustión. Es decir, dada una cantidad de combustible determinada, habrá que garantizar una cierta cantidad de aire fresco de entrada para que el nivel de humos se mantenga por debajo de cierto límite.

De lo anterior se deduce que, en general, la necesidad de evitar la formación de humos limita la capacidad de reducir la emisión de $NO_x$, por lo que habrá

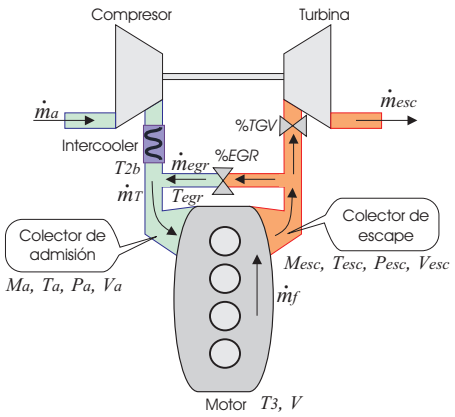


***Figura 7.7:*** Esquema de un motor diesel con turbocompresor.

que llegar a una solución de compromiso. En efecto, si recirculamos gran cantidad de gases de escape al colector de admisión, entrará menor cantidad de aire fresco lo cual, para una misma cantidad de combustible inyectada, disminuirá el ratio $AFR$, con lo que aumentará la cantidad de humo.

Un motor para un vehículo de pasajeros es, por definición, un proceso en el que se producen frecuentemente cambios en variables externas (perfil de la carretera, demandas del conductor, etc... ). Por esta razón, cobra especial importancia no sólo elegir puntos de funcionamiento adecuados para las variables que afectan a los contaminantes, sino también unos controladores que permitan unos transitorios lo más rápidos posible, puesto que durante éstos se producirá una parte importante de las emisiones.

### Estrategia de control propuesta

Vistos los objetivos que se pretenden con el control del motor, parecería lógico plantear el control multivariable, utilizando como actuadores $EGR$ y $TGV$, de las siguientes variables:

**Ratio aire-fuel (AFR)** Cuando esta magnitud está por debajo de un determinado valor, la combustión no se produce correctamente y aparecen humos.

**Fracción de gas quemado ($F_1$)** Se define como el cociente entre densidad de los productos de combustión y total en el colector de admisión. Cuanto más elevado sea, menos $NO_x$ se producen.

No obstante, no se dispone de sensores capaces de medir dichas magnitudes. En su lugar, la información del proceso se obtiene a partir de sensores convencionales que miden la presión en el colector de admisión, $P_a$, y el flujo másico de

aire que circula por el compresor, $\dot{m}_a$. Estas variables están relacionadas con $F_1$ y $AFR$, de forma que es posible obtener un mapa que relacione las referencias de las últimas con las primeras [vNMS+98].

Por otro lado, existen otras dos variables que afectan en gran medida al comportamiento del motor:

**Velocidad del motor ($N$)** Depende del par motor generado, del par resistente y de la inercia del vehículo. El par resistente depende a su vez de las condiciones de la carretera, el peso del vehículo, etc...

**Masa de fuel ($\dot{m}_f$)** Se trata de la cantidad de fuel inyectado. Se controla con un sistema independiente al de renovación de la carga, que es el que nos ocupa. La referencia de masa de fuel deseada es obtenida por un mapa estático en función de $N$, y de la posición del pedal de aceleración. Además, las unidades de control electrónico ($ECU$) de los motores suelen incluir un dispositivo limitador de humos, que limita la cantidad de fuel inyectado cuando el aire que entra en los cilindros no es suficiente, para que no se produzcan humos.

Estas dos variables son las que definen el punto de funcionamiento desde el punto de vista motorístico.

Aunque ninguna de las dos magnitudes es directamente manipulable por el sistema de control de renovación de la carga, ambas son medibles. Por tanto, es conveniente incluirlas de algún modo en el esquema de control.

Si se pretende diseñar un controlador aplicable al proceso real, es importante tener presente que éste se implementa en una unidad de control electrónica (ECU) con unos recursos limitados. En este contexto, una posibilidad es ajustar varios controladores para diferentes valores de $N$ y $\dot{m}_f$ y cambiar entre ellos mediante una estrategia de planificación de ganancia.

Por tanto, en una primera fase, el objetivo es diseñar cada uno de esos controladores. Desde este punto de vista, el problema es controlar un sistema no lineal con dos entradas manipulables ($EGR$ y $TGV$), y dos variables a controlar ($P_a$ y $\dot{m}_a$).

### Control Predictivo

Para realizar el control de este sistema, una propuesta es el control predictivo mediante modelos Hammerstein-Wiener planteado en [PBGNS06]. En dicho trabajo, se mostró que el control predictivo con modelos lineales no permite un ajuste agresivo de los controladores por la baja calidad del modelo de predicción. Además, se descartó un control predictivo basado en programación no lineal en línea por las limitaciones en tiempo de cómputo que implica la implementación en una ECU, por lo que se optó por un modelado mediante sistemas Hammerstein-Wiener como el descrito en la sección 7.3.2.

Para realizar este modelado, en primer lugar se obtuvieron las funciones no lineales $h(EGR, TGV)$ y $w(EGR, TGV)$. En concreto, en dicho trabajo se consideraron como dos alternativas diferentes un modelo de tipo Hammerstein y otro de tipo Wiener. En ambos casos, la función no lineal es obtenida experimentalmente. El experimento consiste en realizar varios ensayos que cubren todo el rango de entradas a intervalos suficientemente bajos y para cada uno de ellos, tomar los valores de $P_a$ y $\dot{m}_a$ en los que el sistema se estabiliza. De esta forma, la no linealidad representa por completo el comportamiento estático del sistema, siendo el sistema lineal identificado forzado a una ganancia unitaria. Las figuras 7.8 y 7.9 representan la no linealidad así identificada. Como se ha

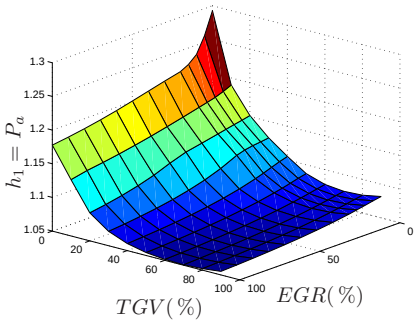


***Figura 7.8:*** No linealidad estática $P_a$.

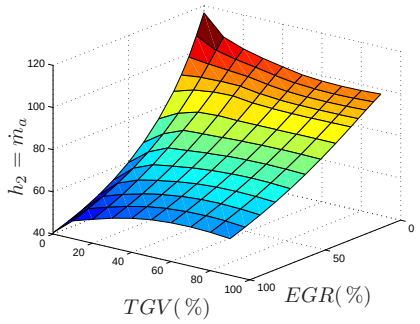


***Figura 7.9:*** No linealidad estática $\dot{m}_a$.

visto en la sección 7.3.2, para poder implementar el controlador mediante cancelación de la no linealidad, es necesario obtener la inversa de ésta y el modelo lineal. En [PBGNS06] se detallan los procedimientos para calcular ambas. Además, se muestra que, entre las dos propuestas analizadas, un modelo de tipo Hammerstein es el que mejor se ajusta al comportamiento del motor.

Una vez obtenida la función no lineal inversa y un modelo lineal definido en incrementos, es posible reescribir las restricciones existentes en el procesos de saturación de las válvulas de $EGR$ y $TGV$ en función de las nuevas variables $v_1$ y $v_2$ mediante las ecuaciones (7.6). La figura 7.10 muestra el espacio de restricciones sobre las nuevas variables que, como puede observarse es no convexo.

En [PBGNS06] se planteó un tratamiento de dichas restricciones sustituyéndolas por un rectángulo de saturaciones exteriores a la región y un *clipping* posterior de las acciones de control reales calculadas, $EGR$ y $TGV$. Aunque los resultados obtenidos fueron bastante aceptables, se detectaron dos inconvenientes fundamentales:

- En algunos transitorios, el controlador daba acciones de control no factibles (antes de realizar el clipping), lo que se traducía en respuestas oscilatorias.
- El esquema propuesto no ofrecía ningún tipo de garantías de estabilidad.

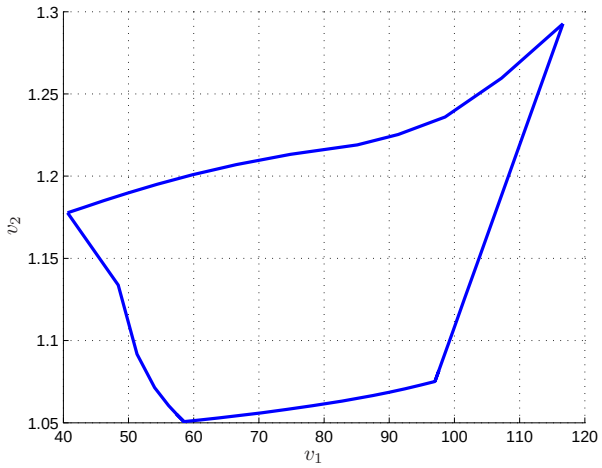


***Figura 7.10:*** Restricciones no lineales.

En este contexto, es interesante reconsiderar el problema con una definición de un espacio de restricciones no convexo interior a la región representada en 7.12 y diseñar un controlador explícito como los propuestos en capítulos anteriores. Con objeto de compararlo, se diseñarán también otros dos controladores predictivos, uno con una región de restricciones interiores convexas como la representada en 7.11 y otra con unas restricciones definidas como la envolvente convexa del conjunto original.

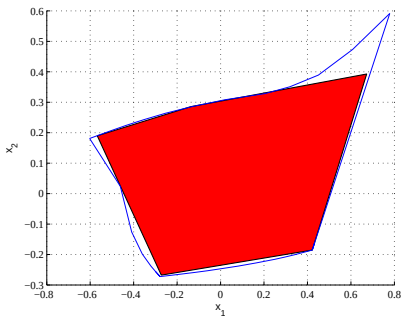


***Figura 7.11:*** Espacio de restricciones interior convexo.

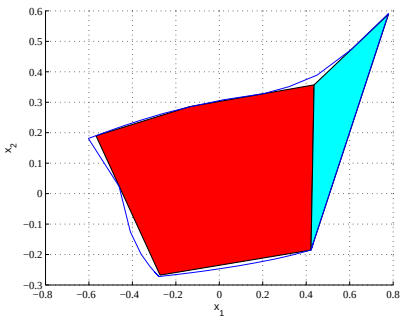


***Figura 7.12:*** Espacio de restricciones no convexo.

Con cada uno de estos conjuntos de restricciones, se definen tres controla-

dores predictivos con los siguientes parámetros:

$$R = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$
$$Q = C^T \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix} C$$
$$N = 2$$

Para calcular la solución explícita del controlador predictivo con restricciones no convexas, definido sobre 4 posibles conjuntos $T_i$, se utiliza la metodología de intersección, división y unión expuesta en la sección 4.3. Tras aplicar el algoritmo de intersección y división, se obtienen 1313 regiones. Una vez se unen las regiones con una misma solución, quedan un total de 1143, sobre las que se aplica el algoritmo de eliminación de soluciones subóptimas. Tras realizar 3434 problemas de suma de cuadrados, se comprueba que pueden eliminarse 3400 soluciones. La aplicación del algoritmo de unión por última vez, proporciona una solución explícita final de 410 regiones, 11 de ellas con 2 soluciones, 2 con 4 soluciones y el resto con una única solución.

Por su parte, las soluciones explícitas con restricción convexa interior y exterior (envolvente convexa) están definidas, respectivamente, por 66 y 52 regiones.

A continuación, se realiza una simulación para comparar el controlador diseñado con las dos alternativas más sencillas propuestas. Para los tres controladores predictivos, dicha simulación consiste en, partiendo de un punto inicial $P_a = 1{,}07\ bar$ y $\dot{m}_a = 46{,}3\ kg/h$ llevar al sistema hasta el punto de equilibrio ($P_a = 1{,}127\ bar$ y $\dot{m}_a = 73{,}8\ kg/h$). Las figuras 7.13 y 7.14 muestran los resultados de la simulación para los tres controladores.

Aunque las respuestas obtenidas para el sistema en los tres casos son bastante similares, pueden observarse como el controlador basado en las restricciones no convexas ofrece una mejor respuesta que las dos alternativas. En el caso del controlador con restricciones convexas internas, puede apreciarse que durante los instantes iniciales no alcanza la saturación de una de las acciones de control (TGV) y esto repercute en una respuesta más lenta del sistema, en especial en $P_a$. En cuanto al controlador definido con la envolvente convexa de las restricciones puede observarse que la respuesta es mucho más próxima a la del controlador con restricciones no convexas puesto que, tras realizar el *clipping* de las acciones de control obtenidas, éstas son muy similares. No obstante, se aprecia un error en la masa de aire que tarda cerca de dos segundos en corregirse. Debido a que, como se ha dicho, el sistema controlado es un proceso en el que los transitorios son frecuentes, la ventaja obtenida con el controlador con restricciones no convexas puede ser significativa.

Los resultados mostrados hasta el momento no han incluido en el diseño de los controladores ninguna región terminal de restricciones y, por lo tanto, no garantizan la estabilidad del sistema en bucle cerrado. A continuación, se diseñan dos nuevos controladores con garantías de estabilidad, uno con las restricciones no convexas y otro con las restricciones convexas interiores. La

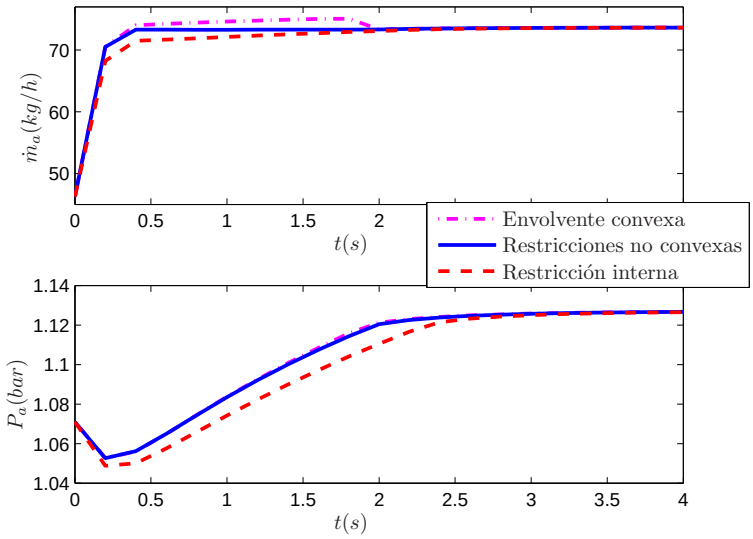


***Figura 7.13:*** Salidas del sistema.

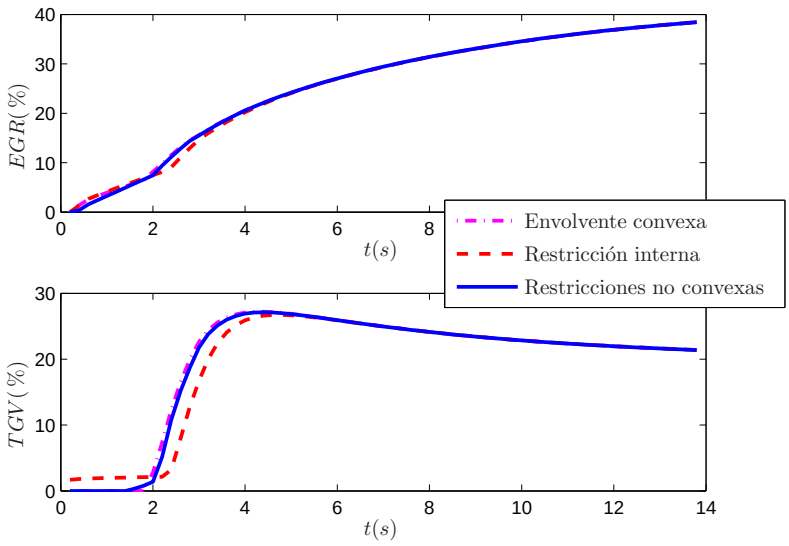


***Figura 7.14:*** Acciones de control.

envolvente convexa de las restricciones, al incluir puntos no factibles, no puede usarse en un esquema en el que se busquen garantías de estabilidad.

Por tanto, para el controlador con restricciones no convexas se diseña una región terminal siguiendo el procedimiento descrito en el capítulo 6 obteniéndose una región definida mediante 4 poliedros convexos. De esta forma, el problema no convexo está definido sobre 16 conjuntos $T_i$. En este caso, la aplicación de los algoritmos de intersección y división, resulta en 4868 regiones. Tras la unión de las que tienen la misma solución, se obtiene un total de 2378 poliedros. Mediante el algoritmo de eliminación de soluciones no óptimas se comprueba, tras la realización de 13326 problemas de suma de cuadrados, que pueden eliminarse 10121 soluciones. Aplicando por última vez el algoritmo de unión, la solución explícita final queda definida mediante 1649 poliedros, 1186 con una única solución y el resto con varias (hasta 12).

Por su parte, el controlador con garantías de estabilidad y restricciones convexas interiores tiene una solución explícita definida por 151 regiones.

En este caso, no se puede repetir la simulación realizada para los controladores anteriores puesto que desde ese punto inicial no hay garantías de estabilidad. En su lugar, se realiza una simulación desde el punto inicial $P_a = 1{,}09\ bar$ y $\dot{m}_a = 28{,}7\ kg/h$ hasta el punto de equilibrio. La respuesta del sistema se compara con la obtenida con el controlador con garantías de estabilidad definido con las restricciones convexas interiores. Las figuras 7.15 y 7.16 muestran los resultados de dicha simulación.

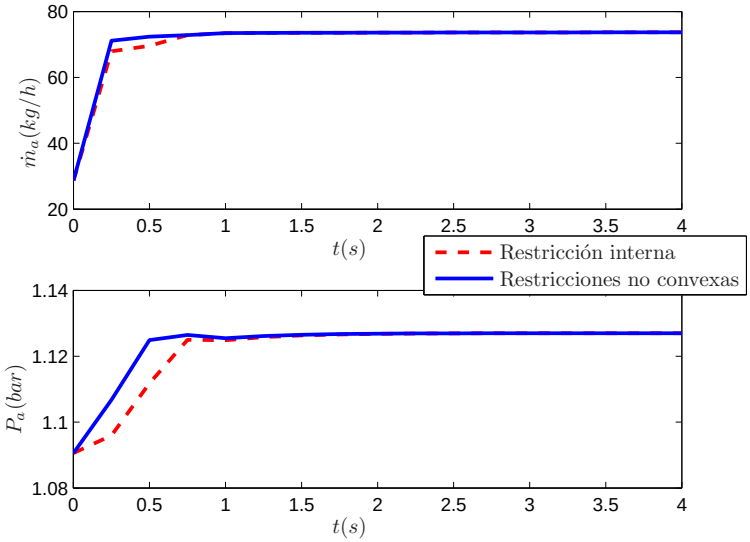


***Figura 7.15:*** Salidas del sistema con controlador estable.

Como en el caso anterior, se observa que la respuesta obtenida con el con-

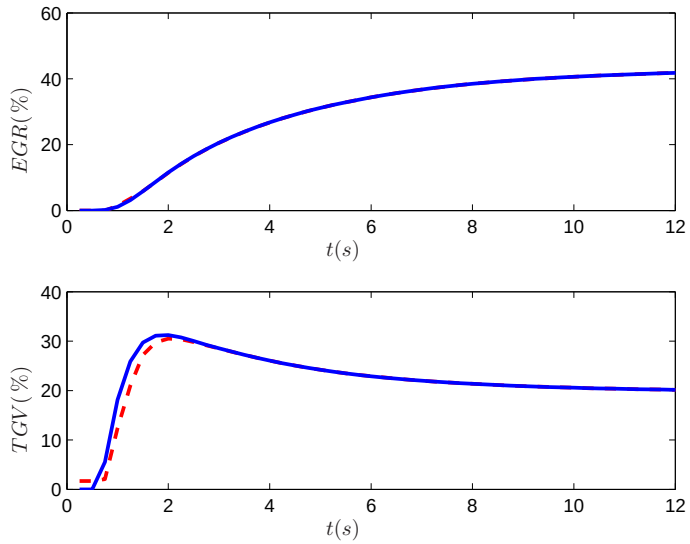


***Figura 7.16:*** Acciones de control con controlador estable.

trolador con restricciones no convexas es más rápida.

Con este diseño de controladores, es interesante también analizar la región de atracción, es decir, el conjunto de estados iniciales desde los que existen garantías de que es posible estabilizar el sistema en bucle cerrado. La figura 7.17 muestra, para los dos controladores propuestos, una sección de dicho espacio (definido en $\mathbb{R}^4$) para los puntos iniciales de $\dot{m}_a$ y $P_a$ en régimen estacionario, escalada en unidades de las salidas del sistema.

Puede comprobarse que el espacio es mayor para el controlador con restricciones no convexas, aunque la diferencia no es demasiado elevada (puesto que la no convexidad de las restricciones originales no es muy acusada).

### 7.3.4. Control Predictivo de sistemas con restricciones mediante linealización por realimentación

En esta sección se estudiará como el control predictivo con restricciones poliédricas no convexas pueden ser también una herramienta de utilidad para el control predictivo mediante linealización entrada-salida por realimentación.

Considérese inicialmente un sistema SISO no lineal descrito por la siguiente ecuación en espacio de estados

$$\dot{x} = f(x) + g(x)u \tag{7.12}$$
$$y = h(x) \tag{7.13}$$

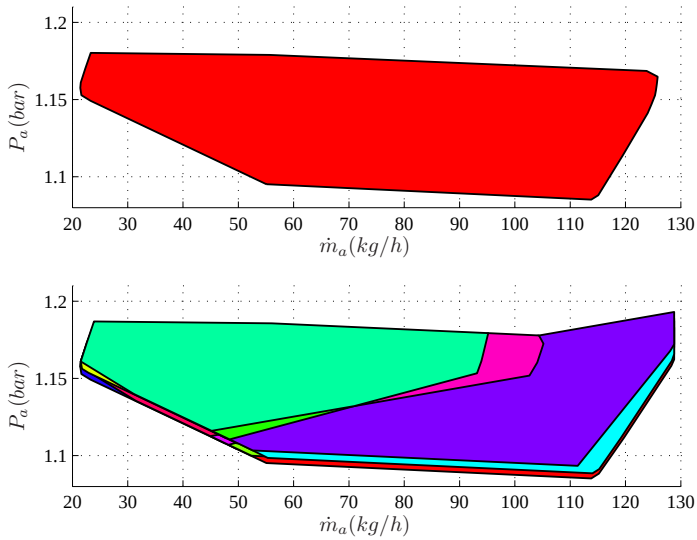


***Figura 7.17:*** Sección del conjunto de estados iniciales factibles.

La idea básica del control mediante linealización entrada-salida por realimentación consiste en buscar una nueva acción de control equivalente $v$ para la que exista una relación lineal con la salida $y$ [SL91]. Para el nuevo sistema, es posible implementar técnicas de control para procesos lineales, como el control predictivo.

Para ello, es de utilidad introducir la siguiente definición:

**Definición 7.1.** Dada una función escalar suave $h : \mathbb{R}^n \to \mathbb{R}$ y un campo vectorial suave $f : \mathbb{R}^n \to \mathbb{R}^n$, la derivada de Lie de $h$ con respecto a $f$ es la función escalar definida por $L_f h = \nabla h\, f$.

Además, la derivada de Lie puede definirse de manera recursiva:

$$L_f^0 h = h$$
$$L_f^i h = L_f(L_f^{i-1}h) = \nabla(L_f^{i-1}h)f \quad \text{para: } i = 1, 2, \ldots$$

El concepto de derivada de Lie $L_f h$ representa la derivada direccional de $h$ en la dirección marcada por $f$ y tiene importancia porque, a partir de (7.12), puede escribirse:

$$\dot{y} = \nabla h\, \dot{x} = \nabla h(f + gu) = L_f h(x) + L_g(x)u$$

Puede observarse que, si existe alguna región $\Omega$ para la que $L_g h(x) \neq 0$, la acción de control $u$ afecta en dicha región a la derivada de la salida. Por tanto, realizando el cambio de variable

$$u = \frac{1}{L_g h}(-L_f h + v)$$

se tiene un sistema lineal entre $y$ y $v$, $\dot{y} = v$.

Si, por el contrario, se tiene $L_g h(x) = 0$ para todo $x$, puede calcularse una nueva derivada y comprobar si $u$ aparece en $\ddot{y} = L_f^2 h(x) + L_g L_f(x) u$. Si se tiene $L_g L_f(x) = 0 \; \forall x$, es necesario seguir derivando hasta que se encuentra un entero $r$, que se define como *grado relativo*, para el que se cumple

$$L_g h L_f^{r-1} h(x) \neq 0$$

para alguna región $\Omega$. Entonces, se define la ley de control

$$u = \frac{1}{L_g h L_f^{r-1} h}(-L_f^r h + v) \tag{7.14}$$

de forma que el sistema tiene la dinámica

$$y^{(r)} = L_f^r h(x) + L_g L_f^{r-1} h(x) u = v$$

claramente lineal con respecto a la entrada equivalente $v$.

Una vez llegados a este punto, es posible en teoría realizar el diseño de un controlador con cualquiera de las técnicas existentes para sistemas lineales. No obstante, en la práctica, existen diferentes inconvenientes que es necesario subsanar.

El primero de dichos inconvenientes aparece debido a la necesidad de un modelo exacto del sistema no lineal, difícil de obtener en la práctica.

El segundo, tiene que ver con el grado relativo del sistema. Es posible demostrar que el nuevo sistema lineal puede escribirse en la llamada forma normal, para la que los primeros $r$ estados vienen dados por la salida y sus derivadas, $\mu = [y \, \dot{y} \ldots y^{(r-1)}]^T$ y el resto están definidos por una dinámica no lineal, $\dot{\psi} = w(\mu, \psi)$. Aunque se diseñe un controlador que haga comportarse al sistema linealizado de una manera adecuada, es necesario comprobar que la dinámica interna, $\dot{\psi} = w(\mu, \psi)$, no se hace inestable. Para realizarse este análisis, se estudia lo que se conoce como *dinámica de los ceros*, que se define como la dinámica interna del sistema cuando la acción de control es tal que la salida $y$ y sus derivadas se mantienen nulas:

$$\begin{aligned} \dot{\mu} &= 0 \\ \dot{\psi} &= w(0, \psi) \end{aligned}$$

Esta doble problemática asociada al control por realimentación entrada-salida ha sido objeto de investigación en el campo, siendo soluciones habituales el control robusto y adaptativo y el análisis sistemático de la dinámica de los ceros.

El concepto de linealización introducido puede extenderse con relativa sencillez al caso MIMO derivando las diferentes salidas $y_i$ hasta que alguna de las entradas aparece. Para ello, se define el grado relativo $r_i$ como el menor entero

para el que al menos una de las entradas aparece en $y_i^{(r_i)}$:

$$y_i^{(r_i)} = L_f^{r_i} h_i + \sum_{j=1}^{m} L_{g_j} L_f^{r_i - 1} h_i u_j$$

con $L_{g_j} L_f^{r_i - 1} h_i(x) \neq 0$ en un entorno $\Omega_i$ para al menos una $j$.
Operando así para todas las salidas se tiene:

$$\begin{bmatrix} y_1^{(r_1)} \\ \vdots \\ y_m^{(r_m)} \end{bmatrix} = \begin{bmatrix} L_f^{r_1} h_1(x) \\ \vdots \\ L_f^{r_m} h_m(x) \end{bmatrix} + E(x)u$$

Si la matriz $E(x)$ es invertible, puede definirse la siguiente transformación de la entrada

$$u = E^{-1} \begin{bmatrix} v_1 - L_f^{r_1} h_1 \\ \vdots \\ v_m - L_f^{r_m} h_m \end{bmatrix} \tag{7.15}$$

para la que se tiene una dinámica del sistema

$$y_i^{(r_i)} = v_i$$

Puede observarse como la ley de control diseñada, además de proporcionar una linealización, realiza un desacoplamiento del sistema, puesto que cada acción de control equivalente afecta únicamente a una salida.

A la problemática existente para el caso SISO, es decir, modelo no exacto y dinámica interna, es necesario añadir en el caso MIMO la invertibilidad de la matriz $E(x)$.

Es relativamente habitual que dicha matriz no pueda invertirse (porque, por ejemplo, tiene una columna de ceros). Para estos casos en que $E$ es singular, existen métodos que permiten generar una linealización de entrada-salida. Destacan entre estos la extensión dinámica, que añade como nuevas entradas las derivadas de las originales, y una inversión del sistema, que deriva nuevas salidas [SL91].

Además de la problemática habitualmente tratada, existe una dificultad adicional que no es considerada tan frecuentemente en la bibliografía, y es la relativa al tratamiento de restricciones en este tipo de sistemas. Las restricciones a la salida no representan en general demasiada dificultad, puesto que la salida del sistema linealizado es la misma. No obstante, restricciones lineales sobre la entrada $u$, que además son las más habituales por las limitaciones de los actuadores, pasan a ser no lineales y dependientes de $x$ sobre la entrada equivalente $v$, dificultando su tratamiento.

Ante un problema con restricciones, una de las opciones a considerar es siempre el control predictivo. La idea básica es aplicar una ley de control de la forma (7.14) ((7.15) para el caso MIMO) y posteriormente diseñar un controlador predictivo para el sistema lineal de entrada $v$ y salida $y$. El problema

fundamental es que, suponiendo un caso SISO, si se tienen restricciones poliédricas sobre la acción de control de la forma:

$$Ru \leq r$$

éstas se transforman en restricciones no lineales sobre $v$ que además dependen del estado $x$

$$R\frac{1}{L_g h L_f^{r-1} h}(-L_f^r h + v) \leq r \tag{7.16}$$

Si bien la acción de control a aplicar $v_k$ es dependiente de $x_k$, que se considera medible, cuando se tiene un horizonte mayor que 1 es necesario introducir el modelo del sistema, por lo que el juego de restricciones sobre acciones de control futuras es cada vez más complejo.

Resolver el problema de manera exacta requeriría la resolución en línea de problemas SQP, costosos computacionalmente. Con objeto de evitarlos, existen diferentes propuestas en la bibliografía que combinan el control predictivo con la linealización por realimentación y que difieren fundamentalmente en el modo de tratar las restricciones. Por un lado, existen propuestas conservativas (que realizan un tratamiento conservativo de las restricciones) como [Boo97] que utiliza control predictivo robusto. Por otro, existen metodologías basadas en obtener las restricciones sobre la acción de control a aplicar $u_k$ (aprovechando que el primer estado $x_k$ es medible) y extenderlas a lo largo de todo el horizonte [KH97]. Este tipo de estrategias, no obstante, pueden llevar al incumplimiento de las restricciones, aunque pueden funcionar para determinados tipos de sistemas no lineales, como las redes neuronales de [DBS09].
Una propuesta diferente es la introducida en [BBKC99], en la que se aproximan las restricciones no lineales alrededor de un $u_0$ (mediante expansión en serie de Taylor) y se resuelve el problema de optimización (mediante QP). Una vez se obtiene una solución, se comprueba si ésta es realmente factible y, en caso contrario, se aplica un procedimiento iterativo que va modificando las restricciones lineales aproximadas hasta obtener una solución que sí lo sea.

Otra posibilidad para resolver el problema planteado, es aproximar las restricciones no lineales mediante una unión de poliedros. Resulta evidente que las restricciones (7.16) pueden escribirse como una serie de desigualdades no lineales genéricas

$$f_i(u, x) \leq 0$$

Si se extienden a todo el horizonte de predicción se obtiene

$$g_i(\mathbf{u}, x) \leq 0$$

Por tanto, es aplicable la aproximación mediante poliedros detallada en la sección 7.3.1 y el control predictivo mediante restricciones poliédricas no convexas.

En este punto es interesante notar que, aunque pueden obtenerse las funciones no lineales dependientes de todas las acciones de control futuras y a partir de ellas los poliedros $T_i$, los algoritmos para la obtención de estos pueden ser bastante complejos por el número elevado de variables. Una alternativa es en

su lugar aproximar las funciones $f_i$ que dependen únicamente de la primera acción de control y posteriormente extender dichas restricciones al futuro, tal y como se plantea en el capítulo 3. Es decir, se trata de obtener los poliedros $S_i$ que cumplan:

$$(u,x) \in \bar{\mathbb{S}} \Rightarrow f_i(u,x) \leq 0 \qquad \forall i$$

### 7.3.5. Ejemplo de aplicación: Péndulo invertido

Se plantea el control de un péndulo invertido cuyo movimiento está regido por las ecuaciones [Can03]:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= f(x) + g(x)u = \frac{g\text{sen}(x_1) - aml x_2^2 \text{sen}(2x_1)/2}{4l/3 - aml cos^2(x_1)} + \frac{-a\cos(x_1)}{4l/3 - aml cos^2(x_1)} u \\ y &= h(x) = x_1 \end{aligned}$$

donde $x_1$ representa el ángulo en radianes del péndulo desde la vertical, $x_2$ la velocidad angular; $g$ es la constante gravitatoria, $m$ la masa del péndulo, $2l$ la longitud del péndulo, $u$ la fuerza en Newtons aplicada al carro y $a = 1/(m+M)$, siendo $M$ la masa del carro.
Para el ejemplo particular, las constantes anteriores toman los valores:

$$\begin{aligned} M &= 8\ kg \\ m &= 2\ kg \\ l &= 2\ m \\ g &= 9{,}8\ m/s^2 \end{aligned}$$

Además, se suponen unas restricciones sobre la acción de control

$$-300 \leq u \leq 300$$

Para aplicar la linealización entrada-salida por realimentación, en primer lugar se calculan las derivadas de Lie:

$$L_g h(x) = \nabla h\ g = [1 \quad 0] \begin{bmatrix} 0 \\ \frac{-a\cos(x_1)}{4l/3 - aml cos^2(x_1)} \end{bmatrix} = 0$$

$$L_f h(x) = \nabla h\ f = [1 \quad 0] \begin{bmatrix} x_2 \\ \frac{g\text{sen}(x_1) - aml x_2^2 \text{sen}(2x_1)/2}{4l/3 - aml cos^2(x_1)} \end{bmatrix} = x_2$$

Dado que $L_g h = 0$ para cualquier $x$, es necesario calcular una derivada de orden mayor:

$$L_g L_f h(x) = \nabla(L_f h)\ g = [0 \quad 1] \begin{bmatrix} 0 \\ \frac{-a\cos(x_1)}{4l/3 - aml\cos^2(x_1)} \end{bmatrix} = \frac{-a\cos(x_1)}{4l/3 - aml\cos^2(x_1)}$$

Como se ha obtenido una derivada de Lie no nula, se tiene que el grado relativo $r = n = 2$ y por tanto el sistema no presenta dinámica interna y la ley de control que linealiza el sistema es:

$$u = \frac{4l/3 - amlcos^2(x_1)}{-a\cos(x_1)} \left( -\frac{g\text{sen}(x_1) - amlx_2^2\text{sen}(2x_1)/2}{4l/3 - amlcos^2(x_1)} + v \right)$$

y el nuevo sistema a controlar:

$$\ddot{y} = v$$

A continuación, para implementar el control predictivo, es necesario trasladar las restricciones sobre la acción de control $u$ a la entrada equivalente $v$. Estas restricciones son ahora no lineales y dependientes de $x$, como muestra la figura 7.18.

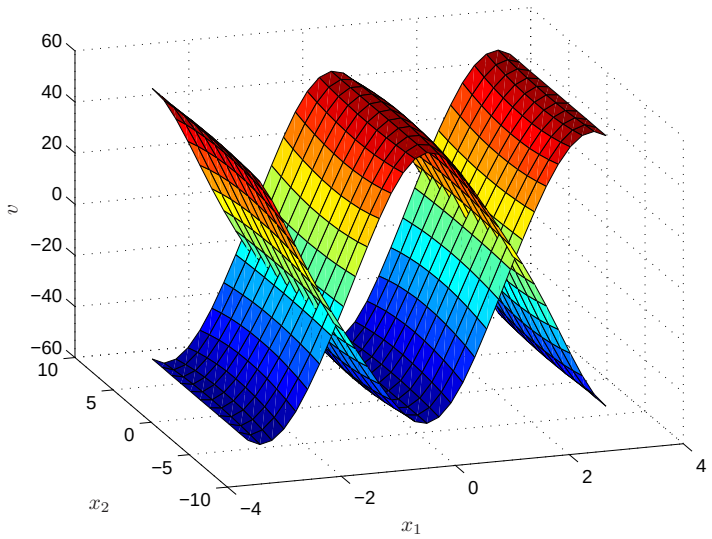


***Figura 7.18:*** Espacio de restricciones no convexo.

Con objeto de aplicar la metodología del control predictivo con restricciones no convexas, se busca una aproximación interior mediante poliedros de las funciones no lineales anteriores, obteniéndose los siguientes poliedros:

$$\bar{\mathbb{S}} = S_1 \cup S_2 \cup S_3$$

$$S_1 = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 1 & 0 \\ 0{,}9993 & 0 & -0{,}0364 \\ 0{,}9997 & 0 & 0{,}0250 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ v \end{bmatrix} \leq \begin{bmatrix} \pi \\ 2\pi \\ 2\pi \\ -1{,}2132 \\ -1{,}8156 \end{bmatrix}$$

$$S_2 = \begin{bmatrix} -0{,}9997 & 0 & -0{,}0250 \\ 0 & -1 & 0 \\ 0 & 1 & 0 \\ -0{,}9993 & 0 & 0{,}0364 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ v \end{bmatrix} \leq \begin{bmatrix} -1{,}8156 \\ 2\pi \\ 2\pi \\ -1{,}2132 \\ \pi \end{bmatrix}$$

$$S_3 = \begin{bmatrix} 0 & -1 & 0 \\ -0{,}9993 & 0 & -0{,}0364 \\ -0{,}9997 & 0 & 0{,}0250 \\ 0 & 1 & 0 \\ 0{,}9997 & 0 & -0{,}0250 \\ 0{,}9993 & 0 & 0{,}0364 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ v \end{bmatrix} \leq \begin{bmatrix} 2\pi \\ 1{,}9263 \\ 1{,}325 \\ 2\pi \\ 1{,}325 \\ 1{,}9263 \end{bmatrix}$$

Como puede apreciarse en la figura 7.19, estas restricciones poliédricas son interiores a las funciones no lineales.

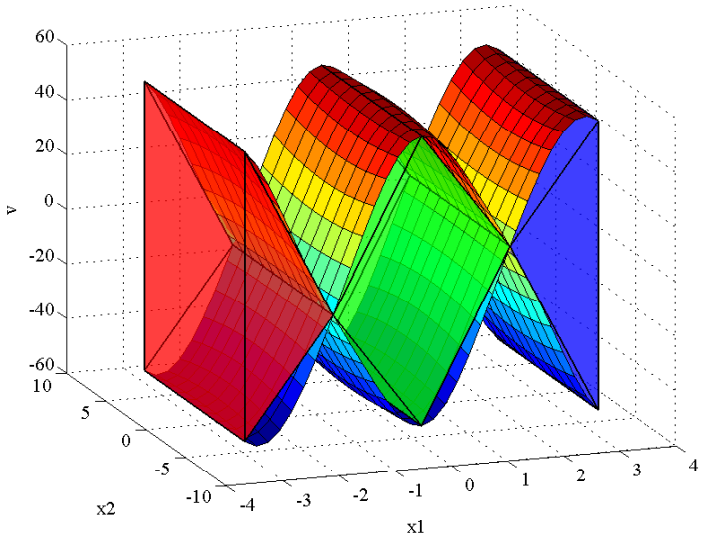


***Figura 7.19:*** Aproximación interior mediante poliedro no convexo.

Para el diseño del controlador predictivo, se discretiza el sistema con un periodo de $T_s = 0{,}05$ segundos y se escogen los siguientes parámetros:

$$R = 1$$
$$Q = \begin{bmatrix} 10 & 0 \\ 0 & 1 \end{bmatrix}$$
$$N = 7$$

Con dichos parámetros se diseña un controlador y región terminales, mediante el procedimiento descrito en el capítulo 6. El controlador obtenido tiene una

ganancia $K = [2{,}9696 \quad 2{,}4371]$ y la región terminal, de un sólo poliedro, es la mostrada en la figura 7.20.

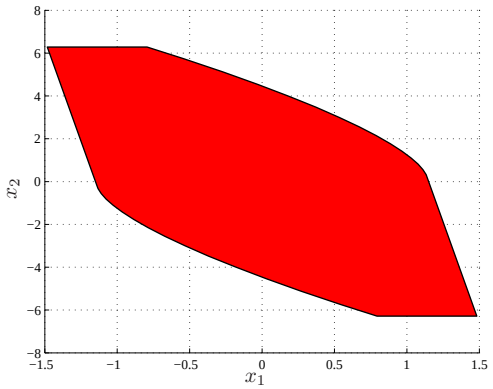


***Figura 7.20:*** Región terminal.

En este punto, y para el horizonte escogido, pueden formarse las restricciones $\mathbb{T}$, definidas por la unión de 15 poliedros $T_i$. Para la obtención de la solución explícita, se aplica la metodología de la envolvente convexa vista en la sección 4.4.
Para ello, en primer lugar se obtiene la solución explícita con las restricciones definidas como la envolvente convexa de todos los $T_i$, obteniéndose 67 regiones. A continuación, se aplican los algoritmos de clasificación y división de regiones, obteniéndose un total de 2010 regiones. Tras unir las que tienen una misma solución, este número se reduce a 1415.
En este punto, se aplica el algoritmo de eliminación de soluciones posibles, obteniéndose, tras la resolución de 2457 problemas de suma de cuadrados, 1055 soluciones que es posible eliminar.
Por último, se aplica de nuevo el algoritmo de unión de regiones hasta obtener las 1084 de la partición final, mostrada en la figura 7.21. De éstas, 111 regiones tienen dos posibles soluciones, 200 tres y 70 cuatro soluciones. El resto, tienen una única solución.

De la figura 7.21, es interesante observar cómo, mediante el propio algoritmo, se han eliminado de los estados iniciales factibles el conjunto de posibles estados desde los que, con las restricciones existentes, no es posible estabilizar el sistema. Así, por ejemplo para el punto $(\pi/2, 0)$ (péndulo en posición horizontal y con velocidad nula), no es posible llevar el sistema a la posición de equilibrio inestable $(0, 0)$.

Por último, se realiza una simulación del sistema no lineal con el controlador diseñado desde la posición de equilibrio estable $(\pi, 0)$ hasta $(0, 0)$. La figura 7.22 muestra la evolución temporal de las variables de estado y la acción de control. Puede observarse como el sistema evoluciona hacia el origen sin violar las restricciones, aunque la acción de control se encuentra muy cercana a la sa-

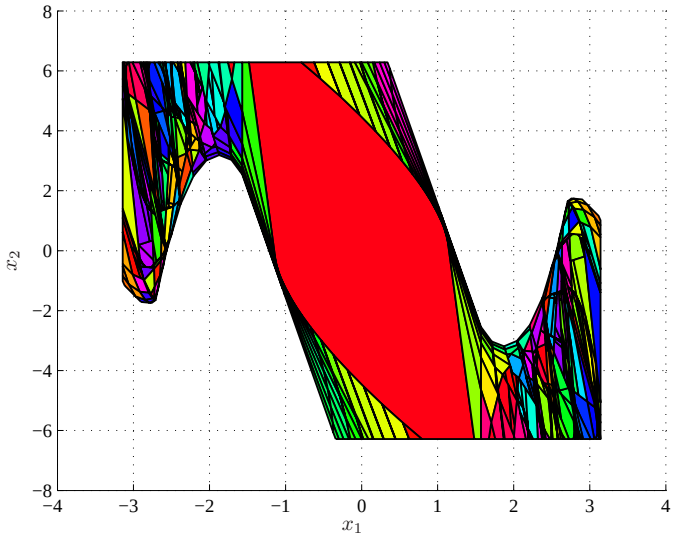


***Figura 7.21:*** Partición final.

turación en algunos instantes. No se produce la saturación porque, como se ha visto, las restricciones poliédricas no convexas utilizadas son una aproximación interior.

Las figuras 7.23 muestra la trayectoria de los estados del sistema y las regiones del controlador explícito calculado que atraviesa.

Por último, la figura 7.24 muestra la trayectoria en $(x, v)$ junto a las restricciones en el espacio de la variable equivalente. En esta figura es interesante observar como el sistema evoluciona de tal forma que, cuando $x_1 = \pi/2$, la acción de control equivalente se hace $v = g$, puesto que, debido a la no linealidad, para cualquier otro valor de ésta la acción de control real $u$ tiende a infinito.

## 7.4. Conclusiones

En este capítulo se han mostrado algunas aplicaciones del control predictivo con restricciones poliédricas no convexas.

En primer lugar, se han introducido algunas de las aplicaciones puras, en las que la propia definición del problema exige el cumplimiento de restricciones poliédricas no convexas. Entre ellas destacan las aplicaciones de evitación de obstáculos y planificación de trayectorias. Además, se ha resuelto el ejemplo de un robot cartesiano que debe evitar una zona poliédrica en su trayectoria.

A continuación, se ha propuesto como aplicación de las técnicas planteadas la aproximación de sistemas con restricciones no lineales. Para ello, se han dado algunas referencias que permiten sustituir restricciones no lineales por

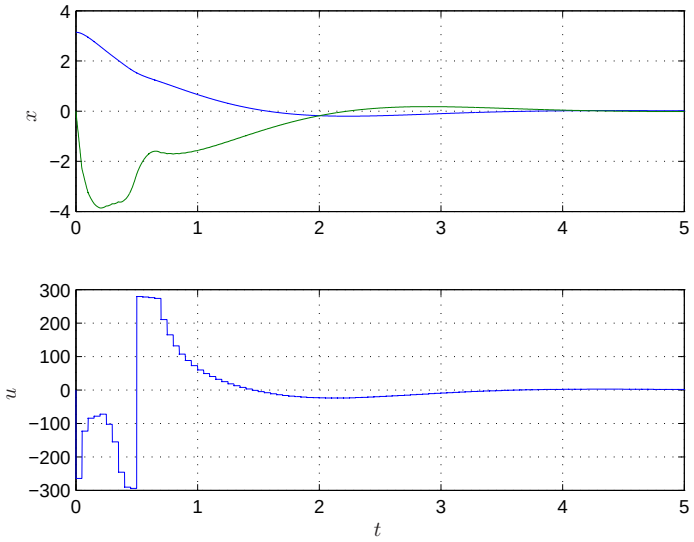


***Figura 7.22:*** Respuesta temporal del sistema.

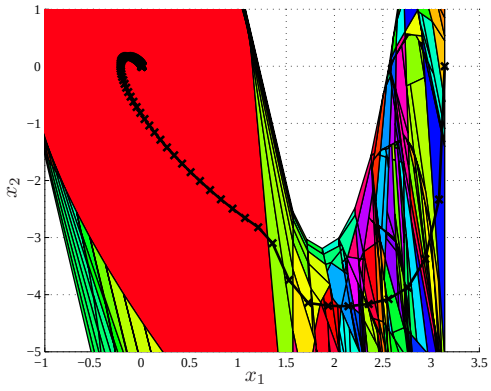


***Figura 7.23:*** Trayectoria de los estados.

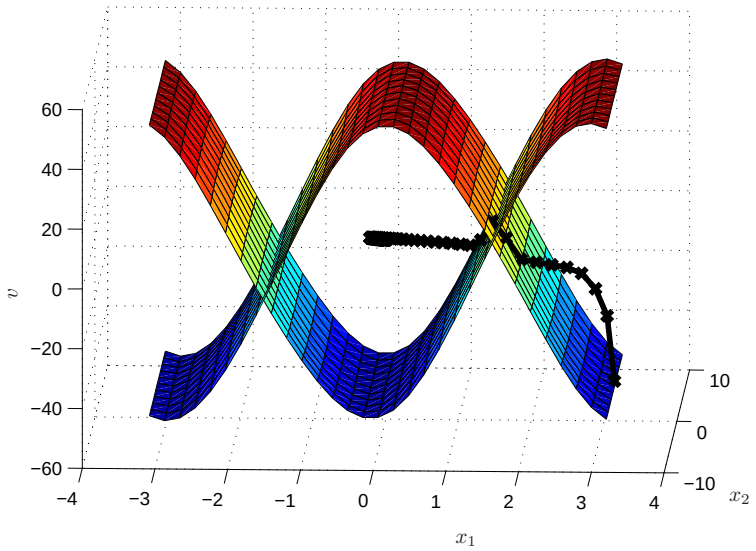


***Figura 7.24:*** Trayectoria en $(x, v)$.

poliedros no convexos interiores a ellas. Entre los sistemas de control para los que aparecen restricciones no lineales, se ha propuesto la cancelación de no linealidades estáticas en sistemas Hammerstein-Wiener y la linealización por realimentación entrada-salida. Se ha resuelto también un ejemplo para cada una de ellas, en concreto el control de la renovación de la carga en un motor Diesel sobrealimentado y el control de un péndulo invertido.

Capítulo 8

# Conclusiones y trabajo futuro

## 8.1. Conclusiones

En la presente tesis doctoral se ha planteado el problema del control predictivo sujeto a restricciones poliédricas no convexas y se ha enfocado su utilización en dos vertientes: como un problema con entidad propia, para problemas como el de evitación de obstáculos, y como una alternativa al control predictivo no lineal cuando períodos de muestreo bajos no permiten la aplicación de programación no lineal.

Las contribuciones de la tesis se pueden dividir en tres ámbitos: obtención de algoritmos en línea eficientes para el cálculo de la solución óptima del problema propuesto, formulación de condiciones suficientes para la estabilidad del sistema en bucle cerrado y aplicación a sistemas no lineales.

### 8.1.1. Algoritmos eficientes para el control predictivo con restricciones poliédricas no convexas

En primer lugar, desde el punto de vista de la eficiencia computacional en línea, se ha justificado la conveniencia de encontrar una solución explícita del problema de optimización frente a la programación no lineal. En cuanto a la existencia y obtención de dicha solución explícita, se han realizado las siguientes aportaciones:

- En el capítulo 3 se ha demostrado la existencia de una solución explícita al problema del control predictivo con restricciones poliédricas no convexas. Dicha solución es afín a tramos definidos mediante desigualdades lineales y cuadráticas.
- Además, se ha demostrado que, para algunos casos concretos, las regiones de la solución tienen una caracterización particular (bien porque están

definidas únicamente por desigualdades lineales, o bien porque las curvas cuadráticas que las caracterizan son semidefinidas positivas).

- En el capítulo 4 se han introducido dos metodologías diferentes para la obtención de la solución explícita al problema de optimización cuadrática con restricciones poliédricas no convexas: la metodología de intersección, división y unión y la de la envolvente convexa.

- La primera de estas metodologías se basa en formular subproblemas con restricciones convexas (cuya unión forma las restricciones originales) y obtener la solución explícita de cada uno de ellos. A continuación, se intersectan las particiones y se analizan las posibles soluciones en cada región resultante.

- Para la intersección de particiones, se han formulado diferentes propiedades y proposiciones para reducir el número de problemas de programación lineal que es necesario resolver.

- Se han introducido procedimientos para determinar cuando es necesaria la división de las regiones poliédricas en función de si más de una de las posibles soluciones es óptima. En primer lugar, se han propuesto métodos basados en programación cuadrática y/o lineal para los casos particulares anteriormente mencionados. En segundo lugar, se ha introducido un procedimiento general, basado en programación de suma de cuadrados, que permite la resolución de la factibilidad de sistemas definidos mediante ecuaciones polinómicas.

- Por último, se han introducido algoritmos para comprobar de manera eficiente si en aquellas regiones para las que se ha averiguado que hay una única solución óptima ésta coincide con la solución de algunas de las regiones contiguas y, en caso afirmativo, si pueden unirse en un número menor de poliedros.

- La segunda metodología planteada se basa en el cálculo de la envolvente convexa de las restricciones. Se ha demostrado que, dado un valor de las variables de estado $x$, el óptimo que se obtiene para un nuevo problema sujeto a unas restricciones definidas como la envolvente convexa de las restricciones originales tiene siempre un coste igual o menor que el óptimo del problema real.

- Usando la propiedad anterior, se ha obtenido la solución explícita del problema simplificado y se han clasificado las regiones en tres grupos diferentes: las que coinciden exactamente con regiones de la solución del problema real, las que están en el interior de alguna de dichas regiones y el resto.

- Se ha demostrado que sólo las regiones del último tipo pueden tener más de un óptimo, y por tanto puede ser necesario dividirlas. De manera análoga a la metodología alternativa planteada, se ha planteado un procedimiento de división, otro de eliminación de soluciones no óptimas

basado en programación de suma de cuadrados y uno de unión para las regiones con una misma expresión afín del óptimo.

- Por último, se han comparado los dos algoritmos planteados en términos de coste computacional fuera de línea. Se ha demostrado que, aunque se trata de un resultado dependiente del problema, para la mayoría de los casos es más conveniente el algoritmo basado en la envolvente convexa.

Una vez conseguida una solución explícita del problema, en el capítulo 5 se han desarrollado algoritmos en línea eficientes para utilizarla. En este aspecto, se han desarrollado las siguientes contribuciones:

- Se han planteado tres algoritmos diferentes, todos ellos basados en la filosofía de los árboles de búsqueda binarios:
  - Árboles binarios de los subproblemas convexos y comparación en línea del índice de coste.
  - Árbol binario de profundidad mínima: la construcción del árbol requiere la resolución de problemas SOS.
  - Árbol binario de profundidad no mínima: árbol binario de una partición lineal y comparación de costes en las regiones en las que sea necesario.
- Se ha realizado un estudio y comparación de los tres métodos propuestos, concluyendo que el árbol binario de profundidad no mínima será el más adecuado en la mayoría de ocasiones puesto que con un coste fuera de línea moderado (no requiere la resolución de ningún SOS) se ha demostrado que ofrece un coste en línea menor que el obtenido con los árboles binarios de los subproblemas. En cuanto al árbol de profundidad mínima, se ha probado que dependiendo de la cantidad de desigualdades cuadráticas que sea necesario evaluar puede tener un coste mayor.
- Considerando el caso en el que un período de muestreo bajo impida la aplicación de estos algoritmos por su coste en línea, se han propuesto dos simplificaciones subóptimas:
  - Simplificación del problema de optimización de la solución: consiste en reemplazar las restricciones no convexas por un poliedro convexo para instantes posteriores a un determinado horizonte. Dicho poliedro se escoge como un poliedro interior a las restricciones o como la envolvente convexa de éstas. En este segundo caso las primeras acciones de control, que son las únicas que se aplican por la filosofía del horizonte móvil, son factibles, pero las últimas pueden no serlo.
  - Simplificación de la solución explícita: consiste en reemplazar curvas cuadráticas por hiperplanos, con la consiguiente reducción del coste de evaluación.
- Para ambos tipos de simplificaciones, se han planteado métodos para el cálculo de la suboptimalidad en que se incurre.

### 8.1.2. Estabilidad del control predictivo con restricciones poliédricas no convexas

En el capítulo 6 se han analizado las condiciones de estabilidad *a priori* para las técnicas de control planteadas y se han realizado fundamentalmente las siguientes aportaciones:

- Se ha demostrado que el índice de coste es continuo en un entorno del origen, lo que permite asegurar que, si se cumplen todas las condiciones de estabilidad, el sistema en bucle cerrado no es únicamente estable en el sentido de Lyapunov, sino asintóticamente estable.
- Se ha probado que el máximo conjunto invariante en bucle cerrado, que es el conjunto de restricciones terminal óptimo, es un poliedro no convexo.
- Se ha introducido un algoritmo existente para el cálculo de este conjunto invariante usado para restricciones convexas, y se ha extendido para el caso de restricciones poliédricas no convexas mediante el uso de propiedades conocidas para el cálculo del conjunto a un paso para poliedros no convexos.
- Se ha propuesto un algoritmo alternativo basado en obtener el máximo conjunto invariante para las envolventes convexas de las restricciones y eliminar convenientemente de éste algunos estados.
- Por último, se ha analizado la eficiencia de los algoritmos en términos de las regiones convexas cuya unión forma el conjunto final. Se ha demostrado como dicho número de regiones es lineal en los parámetros de entrada para el algoritmo propuesto y polinómico y exponencial en dichos parámetros para el algoritmo genérico.

### 8.1.3. Aplicaciones

En el capítulo 7 se han mostrado algunas aplicaciones para el control predictivo con restricciones poliédricas no convexas:

- Se han introducido algunas de las aplicaciones puras, en las que la propia definición del problema exige el cumplimiento de restricciones poliédricas no convexas. Entre ellas destacan las aplicaciones de evitación de obstáculos y planificación de trayectorias.
- Como ejemplo del tipo de aplicaciones anterior, se ha resuelto el control de un robot cartesiano que debe evitar una zona poliédrica en su trayectoria.
- También se ha propuesto la aplicación de las técnicas planteadas al problema del control predictivo con restricciones no lineales. Para ello, se han dado algunas referencias que permiten sustituir restricciones no lineales por poliedros no convexos interiores a ellas.

- Se ha analizado la problemática del control predictivo de sistemas Hammerstein-Wiener con restricciones lineales. Se ha mostrado como una de las técnicas más habituales para su control, la cancelación de las no linealidades estáticas, transforma el problema en el control predictivo de un sistema lineal con restricciones no lineales (y, por tanto, aproximables por poliedros no convexos).
- Mediante esta metodología, se ha planteado y resuelto el control del sistema de recirculación de la carga en un motor Diesel sobrealimentado.
- Se ha introducido el control predictivo con linealización por realimentación de entrada-salida y comprobado como, de nuevo, aparece un sistema con restricciones no lineales.
- Por último, mediante esta técnica de control no lineal y con el tratamiento de las restricciones planteado en la tesis, se ha resuelto el control de un péndulo invertido.

## 8.2. Trabajo futuro

A la vista de los resultados obtenidos en la presente tesis, se concluye que el principal inconveniente de las técnicas presentadas viene ocasionado por su complejidad intrínseca, que crece de manera exponencial con los horizontes de optimización. Este hecho puede limitar su aplicabilidad bien por la implementación en línea (tanto el coste computacional como los requerimientos de memoria), o bien por el coste computacional fuera de línea para la obtención de la solución explícita. Por tanto, sería conveniente trabajar en diferentes líneas que requieran la solución de problemas menos costosos, aunque para ello sea necesario tolerar cierto grado de suboptimalidad. Con este objetivo fundamental, se podría trabajar en los siguientes aspectos:

- En el capítulo 5 se han planteado ya dos propuestas subóptimas: reemplazar las restricciones no convexas por un poliedro convexo para horizontes relativamente lejanos y sustituir las curvas cuadráticas que definen las regiones de la solución explícita por hiperplanos. Tanto para el caso en que se sustituyen las restricciones por la envolvente convexa como para la aproximación de las curvas por hiperplanos, aunque se ha calculado el grado de suboptimalidad en el que se incurre, no se ha garantizado la estabilidad en bucle cerrado, por lo que sería conveniente analizarla o proponer nuevas alternativas.
- En el capítulo 6, con objeto de diseñar controladores estables, se ha buscado como región terminal el máximo conjunto positivamente invariante de entrada admisible, que resulta ser también en el caso general una región poliédrica no convexa, formada por la unión de varios poliedros. Cuanto mayor sea dicho número de poliedros, mayor es el número de conjuntos de restricciones que es necesario considerar y por tanto la complejidad

del problema de optimización a resolver. Para reducir esta complejidad, podría elegirse como región terminal un subconjunto del máximo conjunto invariante definido por la unión de un número menor de poliedros. No obstante, esto reduciría el espacio de estados iniciales desde los que el sistema en bucle cerrado es estable. Sería interesante desarrollar algún procedimiento que permita buscar un compromiso entre la complejidad del problema y el tamaño de la región de estados iniciales factibles.

- En el capítulo 7 se ha propuesto la utilización de las técnicas desarrolladas para el control de sistemas con restricciones no lineales. Para ello, se ha visto que es necesaria la aproximación interior de espacios no lineales por poliedros no convexos. Sería conveniente poder enfocar esta aproximación de una manera más sistemática. En particular, el resultado ideal sería conseguir calcular la región poliédrica no convexa que cubre una mayor cantidad del espacio de restricciones originales para un número de poliedros fijo. De esta forma, se tendría una manera sencilla de reducir la complejidad del problema (el número de poliedros) sacrificando prestaciones del controlador (puesto que se estarían tomando restricciones cada vez más fuertes).

Aparte de la línea fundamental cuyo objetivo es la simplificación de los problemas a resolver, otra posible mejora de las propuestas presentadas sería el planteamiento de la robustez de los controladores propuestos, incluyendo de alguna forma en el diseño el tratamiento de incertidumbres en los modelos y/o perturbaciones.

Para ello, sería interesante el planteamiento de controladores predictivos *min-max*. La filosofía seguida en este tipo de controladores es la búsqueda de las secuencia de acciones de control futuras que minimizan el índice de coste (dependiente de unos parámetros acotados) para el peor caso posible (obtenido como la maximización del coste en función de dichos parámetros).

La obtención de solución explícita para este tipo de controladores predictivos ha sido abordada para el caso de sistemas sujetos a restricciones poliédricas con un índice de coste lineal en [BBM03a]. Para el caso de controladores diseñados mediante un índice de coste cuadrático, como los propuestos en esta tesis, en [RC06] se demuestra la existencia de una solución explícita, aunque no se propone un algoritmo para su obtención. No obstante, en [dlPnARC07] se plantea el problema en la forma de una programación cuadrática multiparamétrica, para la que existen solvers capaces de obtener la solución.

Para el planteamiento del control predictivo min-max con restricciones poliédricas no convexas podría resultar interesante abordar el problema como la resolución de varias optimizaciones *mpQP* como en [dlPnARC07] y aplicar posteriormente los algoritmos propuestos en esta tesis. La mayor dificultad que se prevé con dicho enfoque es la complejidad de los algoritmos, puesto que el problema min-max se caracteriza por obtener particiones con un elevado número de regiones (creciente con el número de vértices que definen el espacio de perturbaciones). Por ello, podría ser útil también la inclusión de las propuestas

de [ARC05] en las que se simplifica la maximización a resolver mediante un algoritmo de rechazo de vértices.

Apéndice A

# Coste computacional de algoritmos fuera de línea

## A.1. Introducción

En el capítulo 4 se estudia la obtención de la solución explícita del problema de optimización cuadrática con restricciones poliédricas no convexas y, para ello, se plantean diferentes algoritmos. Para el primero de ellos, de intersección, división y unión, se plantean dos posibilidades, un algoritmo simultáneo, A.1, y otro progresivo, A.2.

En el algoritmo A.1 la idea es obtener todas las regiones lineales factibles, comprobando la intersección de todas las combinaciones de regiones de las $\gamma$ particiones y, para cada una de ellas, obtener las curvas que indican cuando cada solución es mejor que las restantes. A priori, para cada una de las regiones lineales hay $\gamma$ posibles soluciones, pero puede que alguna de ellas se pueda descartar porque siempre sea mejor alguna de las otras soluciones.

En el algoritmo A.2 por contra, se parte de dos particiones y se aplica el algoritmo 4.1, válido para $\gamma = 2$. Como resultado de éste se obtiene una nueva partición lineal del estado ($\mathbb{T}_2$) para la que algunas regiones (pertenecientes a $\mathbb{T}_2^1$) tienen una única solución afín, mientras que para otras (las de $\mathbb{T}_2^2$) existen dos posibles soluciones, dependiendo de a qué lado de la correspondiente curva se encuentre el vector de estados $x$. Sobre esta partición $\mathbb{T}_2$, se añade una nueva partición lineal, intersectando sus regiones. En este caso se obtendrán regiones con 1, 2 o hasta 3 posibles soluciones divididas por las correspondientes curvas. Añadiendo de manera sucesiva las $\gamma$ particiones, se llega a la solución explícita final.

Si bien todos los métodos van a dar con la misma solución explícita final, puesto que ésta es única, utilizar uno u otro va a suponer un coste del algoritmo fuera de línea que puede ser significativamente diferente, por lo que, a

**Algoritmo A.1** Obtención de la solución explícita $\gamma > 2$. Algoritmo simultáneo

1. Obtener la solución explícita de los $\gamma$ problemas de optimización con restricciones convexas.
2. Intersectar todas las regiones de todas las particiones.
3. Para cada intersección no vacía, obtener las curvas correspondientes que dividen en varias subregiones.
4. Comprobar si en la región lineal hay más de una solución óptima y por tanto es necesario dividirlas.
5. Comprobar si es posible unir regiones no divididas (con una única solución).

**Algoritmo A.2** Obtención de la solución explícita $\gamma > 2$. Algoritmo progresivo

1. Obtener la solución explícita de los $\gamma$ problemas de optimización.
2. Tomar la partición con menor número de regiones e inicializar:
   - $\mathbb{T}_1^1 = \{X_{1i}\}$, $\mathbb{T}_1^2 = \ldots = \mathbb{T}_1^\gamma = \{\emptyset\}$.
   - $\mathbb{T}_1 = \mathbb{T}_1^1$.
   - $i = 2$.
3. Tomar la siguiente partición con menor número de regiones, $\mathbb{X}_i$.
4. Intersectar con las regiones de cada una de las particiones anteriores: $\mathbb{W}_i^j = \{X_{jk} \cap X_{il} | X_{jk} \in \mathbb{T}_{i-1}^j, X_{il} \in \mathbb{X}_i\}$.
5. Para cada región de $\mathbb{W}_i^j$,comprobar cuáles de las $j+1$ posibles soluciones son aplicables a la región.
6. Actualizar las listas de regiones, $\mathbb{T}_i^j$ en función del número de soluciones aplicables.
7. Si es posible, unir las regiones de $\mathbb{T}_i^j$ con el mismo conjunto de soluciones.
8. $i = i + 1$.
9. Si $i \leq \gamma$, volver a 3.

continuación, se analiza dicho coste para cada uno de los algoritmos propuestos.

Dado que los algoritmos descritos son bastante complejos, resulta complicado analizar su coste en términos de operaciones aritméticas elementales. En lugar de ello, se cuantificará el número de problemas de optimización de diferentes tipos que es necesario resolver. Aunque el tiempo que se tarda en resolver cada problema de un mismo tipo no es constante (puesto que depende de muchos factores como el número de variables, número de restricciones, etc...) sí que se mantiene dentro de un mismo orden de magnitud, y por lo tanto puede ser suficiente para comparar los diferentes algoritmos.

## A.2. Coste del algoritmo simultáneo

En el algoritmo A.1, los pasos que requieren resolver algún problema de optimización, y por tanto son más costosos computacionalmente, son la intersección de las regiones lineales y la comprobación de qué soluciones son posibles en las regiones intersectadas resultantes.
El primero de ambos problemas puede resolverse con una serie de problemas de programación lineal, uno para cada posible región lineal:

$$C_{LP}^{1} = \prod_{i=1}^{\gamma} N_i \tag{A.1}$$

donde $N_i$ es el número de regiones de cada partición.

En cuanto a la comprobación de si una determinada solución de las $\gamma$ posibles es aplicable, como se ha visto ésta puede realizarse mediante la resolución de un problema SOS. Como en este caso en cada una de las regiones lineales existen $\gamma$ posibles soluciones, es necesario resolver $\gamma$ problemas SOS.

Suponiendo que sólo una fracción $\eta$ de los LPs resueltos tengan solución, el número total de problemas SOS a resolver será:

$$C_{SOS}^{1} = \gamma\eta \prod_{i=1}^{\gamma} N_i \tag{A.2}$$

## A.3. Coste del algoritmo progresivo

En cuanto al algoritmo progresivo A.2, éste es algo más complejo de analizar, puesto que el número de problemas de optimización a resolver no es conocido a priori, sino que depende de cuántos de los problemas que se van resolviendo a lo largo de la ejecución del procedimiento tienen solución. Por esta razón, como simplificación inicial se ignorará el paso 7 del algoritmo, suponiendo que la unión de regiones lineales no divididas por curvas no es posible. Más adelante se volverá sobre este aspecto.

Supóngase que se tienen $\gamma$ particiones, cada una de ellas con $N_i$ regiones lineales ($\#\mathbb{X}_i = N_i$).
Al inicio de cualquier iteración $i$ del algoritmo A.2 se dispone de dos particiones del espacio de estados: la que se arrastra de pasos anteriores, $\mathbb{T}_{i-1}$, y la que se desea añadir en la iteración actual, $\mathbb{X}_i$. Las regiones de la primera de dichas particiones están agrupadas en subconjuntos de regiones, $\mathbb{T}_{i-1}^j$, cada uno de ellos con $j$ soluciones afines posibles. Obviamente, la unión de dichos subconjuntos forma la partición original:

$$\mathbb{T}_i = \bigcup_{j=1}^{i} \mathbb{T}_i^j \tag{A.3}$$

El primer paso a realizar en cada iteración es intersectar todas las regiones de cada una de dichas particiones. Se define como $\mathbb{W}_i$ al conjunto de regiones factibles obtenidas de las intersecciones descritas, y $\mathbb{W}_i^j$ al conjunto de regiones factibles obtenidas de intersectar la partición actual, $\mathbb{X}_i$ con cada una de las regiones de los distintos $\mathbb{T}_{i-1}^j$. Nótese que así como $\mathbb{T}_i$ y $\mathbb{W}_i$ son particiones del espacio de estados, $\mathbb{T}_i^j$ y $\mathbb{W}_i^j$ no lo son, puesto que la unión de todas sus regiones no forman el espacio completo.
Por definición se tiene que:

$$\mathbb{W}_i = \mathbb{T}_{i-1} \cap \mathbb{X}_i = (\bigcup_{j=1}^{i-1} \mathbb{T}_{i-1}^j) \cap \mathbb{X}_i = \bigcup_{j=1}^{i-1} (\mathbb{T}_{i-1}^j \cap \mathbb{X}_i) = \bigcup_{j=1}^{i-1} \mathbb{W}_i^j \tag{A.4}$$

Por otra parte, es evidente que $\#\mathbb{W}_i = \#\mathbb{T}_i$. Eso es así porque el conjunto $\mathbb{W}_i$ contiene todas las regiones que son factibles en la iteración anterior, agrupadas en diferentes subconjuntos dependiendo de cómo han sido obtenidas. Por su parte, el conjunto $\mathbb{T}_i$ contiene esas mismas regiones factibles, pero agrupadas en diferentes subconjuntos dependiendo de el número de soluciones que tienen.

Se define el parámetro $\eta_i$ como el cociente entre el número de regiones factibles en $\mathbb{W}_i$ y el número de regiones posibles en dicho conjunto:

$$\eta_i = \frac{\#\mathbb{W}_i}{(\#\mathbb{T}_{i-1})(\#\mathbb{X}_i)} \tag{A.5}$$

Definido de esta forma, se tiene $0 \leq \eta_i \leq 1$.

En la iteración inicial, se parte de un $\mathbb{T}_{i-1}$ equivalente a todo el espacio de estados, puesto que no está particionado todavía. Por tanto, $\mathbb{W}_1 = \mathbb{X}_1$ y $\eta_1 = 1$. Se tiene además que $\mathbb{T}_1^1 = \mathbb{T}_1$ puesto que todas las regiones de la partición $\mathbb{T}_1$ tienen una única solución asociada, la correspondiente a $\mathbb{X}_1$. Obviamente en esta iteración inicial no es necesario resolver ninguna optimización.

En la segunda iteración, sobre la partición existente $\mathbb{T}_1$ se añade otra ($\mathbb{X}_2$) cuyas $N_2$ regiones se intersectan con las anteriores. Dado que en la iteración anterior había un único subconjunto de $\mathbb{T}_1$, en ésta tendremos también un único

subconjunto de regiones de intersección: $\mathbb{W}_2 = \mathbb{W}_2^1$. Para calcularlo, es necesario resolver $N_{LP}^2 = \eta_1 N_1 N_2 = N_1 N_2$ LPs, de los cuales tienen solución:

$$\#\mathbb{W}_2 = \eta_1 \eta_2 N_1 N_2$$

El siguiente paso del algoritmo, consiste en comprobar cuáles de esas regiones tienen dos soluciones afines y cuáles sólo una, resolviendo un problema SOS para cada una de ellas. Por tanto, el número total de problemas SOS que es necesario resolver es:

$$N_{SOS}^2 = \eta_1 \eta_2 N_1 N_2$$

Una vez resueltos los problemas SOS, es posible saber cuántas soluciones tienen las regiones de $\mathbb{W}_2$ y por tanto a qué subconjunto de $\mathbb{T}_2$ deben ser asignadas. Definimos $\epsilon_{211}$ como la fracción de problemas SOS que no tienen solución, y $\epsilon_{212}$ a las que sí que la tienen. De esta forma, se tienen divididas todas las regiones de la partición $\mathbb{T}_2$ en dos subconjuntos, dependiendo del número de soluciones posibles:

$$\#\mathbb{T}_2^1 = \epsilon_{211} \eta_1 \eta_2 N_1 N_2$$
$$\#\mathbb{T}_2^2 = \epsilon_{212} \eta_1 \eta_2 N_1 N_2$$

Definimos el factor $\zeta_{ij}$, que representa la fracción de regiones en la iteración $i$ con $j$ soluciones posibles:

$$\zeta_{ij} = \frac{\#\mathbb{T}_i^j}{\#\mathbb{T}_i} \tag{A.6}$$

Evidentemente, teniendo en cuenta (A.3), para cualquier iteración $i$ se tiene:

$$\sum_{j=1}^{i} \zeta_{ij} = 1$$

Para esta segunda iteración en concreto, se tiene:

$$\begin{aligned} \zeta_{21} &= \epsilon_{211} \\ \zeta_{22} &= \epsilon_{212} = 1 - \epsilon_{211} \end{aligned}$$

En la tercera iteración del algoritmo se añade una nueva partición, $\mathbb{X}_3$, con $N_3$ regiones. De nuevo, el primer paso necesario será obtener los conjuntos $\mathbb{W}_3^j$, comprobando cuáles de estas regiones tienen intersección con las regiones lineales existentes hasta el momento (las pertenecientes a $\mathbb{T}_2^1$ y $\mathbb{T}_2^2$). Esto implica la resolución de $N_{LP}^3 = \eta_1 \eta_2 N_1 N_2 N_3$ LPs y se obtiene:

$$\#\mathbb{W}_3 = \eta_1 \eta_2 \eta_3 N_1 N_2 N_3$$

Definimos en este punto el parámetro $\eta_{ij}$ a partir del número de elementos presentes en $\mathbb{W}_i^j$:

$$\eta_{ij} = \frac{\#\mathbb{W}_i^j}{(\#\mathbb{T}_{i-1}^j)(\#\mathbb{X}_i)} \tag{A.7}$$

Existe una relación entre los factores $\eta_{ij}$ y $\eta_i$ definido anteriormente. A partir de (A.4) se tiene:

$$\#\mathbb{W}_i = \sum_{j=1}^{i-1} \mathbb{W}_i^j$$

Substituyendo (A.5) y (A.7):

$$\begin{aligned} \eta_i(\#\mathbb{T}_{i-1})(\#\mathbb{X}_i) &= \sum_{j=1}^{i-1} \eta_{ij}(\#\mathbb{T}_{i-1}^j)(\#\mathbb{X}_i) \\ \eta_i(\#\mathbb{T}_{i-1}) &= \sum_{j=1}^{i-1} \eta_{ij}(\#\mathbb{T}_{i-1}^j) \end{aligned}$$

Substituyendo ahora (A.6):

$$\begin{aligned} \eta_i(\#\mathbb{T}_{i-1}) &= \sum_{j=1}^{i-1} \eta_{ij}\zeta_{(i-1)j}(\#\mathbb{T}_{i-1}) \\ \eta_i &= \sum_{j=1}^{i-1} \eta_{ij}\zeta_{(i-1)j} \end{aligned} \tag{A.8}$$

Para esta tercera iteración se tienen dos conjuntos de regiones: los correspondientes a intersectar las regiones de la nueva partición $\mathbb{X}_3$ con las regiones pertenecientes en la iteración anterior a $\mathbb{T}_2^1$ y a $\mathbb{T}_2^2$ ($\mathbb{W}_3^1$ y $\mathbb{W}_3^2$ respectivamente). El número de elementos en cada una de estas regiones será:

$$\#\mathbb{W}_3^1 = \eta_{31}\zeta_{21}\eta_1\eta_2 N_1 N_2 N_3 \tag{A.9a}$$
$$\#\mathbb{W}_3^2 = \eta_{32}\zeta_{22}\eta_1\eta_2 N_1 N_2 N_3 \tag{A.9b}$$

A continuación es necesario comprobar, para las regiones de ambos subconjuntos, si tienen más de una solución posible:

- Las regiones de $\mathbb{W}_3^1$ pueden tener o bien una única solución, perteneciendo por tanto a $\mathbb{T}_3^1$, o dos, perteneciendo a $\mathbb{T}_3^2$. Para decidirlo, es necesario resolver un problema SOS para cada región.
- Las regiones de $\mathbb{W}_3^2$ pueden tener una, dos o tres, soluciones. Para comprobarlo, para cada región es necesario resolver 3 problemas SOS.

El número de problemas SOS totales en está iteración será por tanto:

$$N_{SOS}^3 = (\#\mathbb{W}_3^1) + 3(\#\mathbb{W}_3^2)$$

Teniendo en cuenta (A.9) tenemos:

$$N_{SOS}^3 = \left(\frac{\eta_{31}\zeta_{21}}{\eta_3} + 3\frac{\eta_{32}\zeta_{22}}{\eta_3}\right)\eta_1\eta_2\eta_3 N_1 N_2 N_3$$

Tras la resolución de los problemas SOS, es posible decidir cómo dividir las regiones de $\mathbb{W}_3^1$ entre $\mathbb{T}_3^1$ y $\mathbb{T}_3^2$ y las de $\mathbb{W}_3^2$ entre $\mathbb{T}_3^1$, $\mathbb{T}_3^2$ y $\mathbb{T}_3^3$. Para ello, definimos el coeficiente $\epsilon_{ijk}$ como la fracción de regiones de $\mathbb{W}_i^j$ con $k$ posibles soluciones. Nuevamente se cumple:

$$\sum_{k=1}^{j+1} \epsilon_{ijk} = 1$$

Es decir, para la iteración actual, tenemos que de $\mathbb{W}_3^1$ se obtendrán $\epsilon_{311}(\#\mathbb{W}_3^1)$ regiones con una única solución y $\epsilon_{312}(\#\mathbb{W}_3^2)$ con dos. Por otro lado, de $\mathbb{W}_3^2$ se obtendrán $\epsilon_{321}(\#\mathbb{W}_3^2)$ con una solución, $\epsilon_{322}(\#\mathbb{W}_3^2)$ con dos y $\epsilon_{323}(\#\mathbb{W}_2^2)$ con tres. Agrupando términos y substituyendo (A.9) se obtiene:

$$\begin{aligned}
\#\mathbb{T}_3^1 &= (\epsilon_{311}\eta_{31}\zeta_{21} + \epsilon_{321}\eta_{32}\zeta_{22})\eta_1\eta_2 N_1 N_2 N_3 \\
\#\mathbb{T}_3^2 &= (\epsilon_{312}\eta_{31}\zeta_{21} + \epsilon_{322}\eta_{32}\zeta_{22})\eta_1\eta_2 N_1 N_2 N_3 \\
\#\mathbb{T}_3^3 &= \epsilon_{323}\eta_{32}\zeta_{22}\eta_1\eta_2 N_1 N_2 N_3
\end{aligned}$$

De las ecuaciones anteriores, es posible deducir:

$$\begin{aligned}
\zeta_{31} &= \frac{\epsilon_{311}\eta_{31}\zeta_{21} + \epsilon_{321}\eta_{32}\zeta_{22}}{\eta_3} \\
\zeta_{32} &= \frac{\epsilon_{312}\eta_{31}\zeta_{21} + \epsilon_{322}\eta_{32}\zeta_{22}}{\eta_3} \\
\zeta_{33} &= \frac{\epsilon_{323}\eta_{32}\zeta_{22}}{\eta_3}
\end{aligned}$$

Si se analiza una cuarta iteración, es sencillo comprobar que el número de LPs a resolver será:

$$N_{LP}^4 = \prod^3 (\eta_i N_i) N_4$$

Esto nos dará unos conjuntos de regiones de intersección con los siguientes elementos:

$$\begin{aligned}
\#\mathbb{W}_4^1 &= \frac{\eta_{41}\zeta_{31}}{\eta_4} \prod^4 (\eta_i N_i) \\
\#\mathbb{W}_4^2 &= \frac{\eta_{42}\zeta_{32}}{\eta_4} \prod^4 (\eta_i N_i) \\
\#\mathbb{W}_4^3 &= \frac{\eta_{43}\zeta_{33}}{\eta_4} \prod^4 (\eta_i N_i)
\end{aligned}$$

Y, por tanto, un número de SOS a resolver de:

$$\begin{aligned}
N_{SOS}^4 &= (\#\mathbb{W}_4^1) + 3(\#\mathbb{W}_4^2) + 4(\#\mathbb{W}_4^3) = \\
&= \left( \frac{\eta_{41}\zeta_{31}}{\eta_4} + 3\frac{\eta_{42}\zeta_{32}}{\eta_4} + 4\frac{\eta_{43}\zeta_{33}}{\eta_4} \right) \prod^4 (\eta_i N_i)
\end{aligned}$$

La tabla A.1 resume el número de regiones en los diferentes subconjuntos y de los problemas de optimización de diferente tipo a resolver para las cuatro primeras iteraciones del algoritmo progresivo.

A la vista de los resultados obtenidos, se puede deducir el número de problemas LP y SOS a resolver en una iteración cualquiera del algoritmo:

$$\begin{aligned} N^i_{LP} &= N_i \prod_{j=1}^{i-1} \eta_i N_i \\ N^i_{SOS} &= \left( \frac{\eta_{i1}\zeta_{(i-1)1}}{\eta_i} + \sum_{j=2}^{i-1} (j+1) \frac{\eta_{ij}\zeta_{(i-1)j}}{\eta_i} \right) \prod_{j=1}^{i} (\eta_j N_j) \end{aligned}$$

Teniendo en cuenta (A.8):

$$N^i_{SOS} = \left( 1 + \sum_{j=2}^{i-1} j \frac{\eta_{ij}\zeta_{(i-1)j}}{\eta_i} \right) \prod_{j=1}^{i} (\eta_j N_j)$$

Por tanto, el coste total del algoritmo, en términos de problemas LP y SOS es:

$$C^2_{LP} = \sum_{i=2}^{\gamma} N_i \prod_{j=1}^{i-1} \eta_i N_i \tag{A.10}$$

$$C^2_{SOS} = \sum_{i=2}^{\gamma} \left( 1 + \sum_{j=2}^{i-1} j \frac{\eta_{ij}\zeta_{(i-1)j}}{\eta_i} \right) \prod_{j=1}^{i} (\eta_j N_j) \tag{A.11}$$

## A.4. Comparación de los algoritmos

Los costes obtenidos para los dos algoritmos dependen del número de regiones en las que está dividida cada una de las particiones ($N_i$) así como de una serie de parámetros ($\eta_i$, $\eta_{ij}$, $\zeta_{ij}$) que han sido definidos para expresar qué cantidad de una serie de problemas de optimización tienen solución factible. Por lo tanto, a priori no es posible comparar con absoluta certeza ambos costes, puesto que dependiendo de cada instancia particular del problema, puede tener un menor coste un procedimiento o el otro. No obstante, como se verá a continuación sí es posible obtener algunas conclusiones respecto a qué valores límite de los parámetros definidos harán mejor un algoritmo u otro.

En primer lugar, se procederá a comparar el número de problemas LP a resolver en cada caso analizando las expresiones (A.1) y (A.10). Por comparación directa de ambos índices es fácil ver que ninguno de los dos va a ser mejor que el otro en cualquier circunstancia. En un extremo, si los diferentes $\eta_i$ son muy próximos a 0 (a excepción de $\eta_1 = 1$) $C^2_{LP}$ va a ser muy próximo a $N_1 N_2$ y por tanto mucho menor a $C^1_{LP}$. En el otro extremo, si los valores de $\eta_i$ son muy

| **i** | $\#\mathbb{T}_i$ | $\#\mathbb{T}_i^1$ | $\#\mathbb{W}_i^1$ | $\#\mathbb{T}_i^2$ | $\#\mathbb{W}_i^2$ | $\#\mathbb{T}_i^3$ | $\#\mathbb{W}_i^3$ | **#probs. de opt.** |
|---|---|---|---|---|---|---|---|---|
| 1 | $N_1$ | - | - | - | - | - | - | 0 LPs |
| | $\eta_1 N_1$ | - | $\eta_1 N_1$ | - | - | - | - | 0 SOS |
| | $\eta_1 N_1$ | $\eta_1 N_1$ | - | - | - | - | - | - |
| 2 | $\eta_1 N_1 \cap N_2$ | - | - | - | - | - | - | $\eta_1 N_1 N_2$ LPs |
| | $\eta_1 \eta_2 N_1 N_2$ | - | $\eta_1 \eta_2 N_1 N_2$ | - | - | - | - | $\eta_1 \eta_2 N_1 N_2$ SOS |
| | $\eta_1 \eta_2 N_1 N_2$ | $\zeta_{21} \prod^2 (\eta_i N_i)$ | - | $\zeta_{22} \prod^2 (\eta_i N_i)$ | - | - | - | - |
| 3 | $\prod^2 (\eta_i N_i) \cap N_3$ | - | - | - | - | - | - | $\eta_1 \eta_2 N_1 N_2 N_3$ LPs |
| | $\prod^3 (\eta_i N_i)$ | - | $\frac{\eta_{31}\zeta_{21}}{\eta_3} \prod^3 (\eta_i N_i)$ | - | $\frac{\eta_{32}\zeta_{22}}{\eta_3} \prod^3 (\eta_i N_i)$ | - | - | $(\#\mathbb{W}_3^1 + 3\#\mathbb{W}_3^2)$ SOS |
| | $\prod^3 (\eta_i N_i)$ | $\zeta_{31} \prod^3 (\eta_i N_i)$ | - | $\zeta_{32} \prod^3 (\eta_i N_i)$ | - | $\zeta_{33} \prod^3 (\eta_i N_i)$ | - | - |
| 4 | $\prod^3 (\eta_i N_i) \cap N_4$ | - | - | - | - | - | - | $\prod^3 (\eta_i N_i) N_4$ LPs |
| | $\prod^4 (\eta_i N_i)$ | - | $\frac{\eta_{41}\zeta_{31}}{\eta_4} \prod^4 (\eta_i N_i)$ | - | $\frac{\eta_{42}\zeta_{32}}{\eta_4} \prod^4 (\eta_i N_i)$ | - | $\frac{\eta_{43}\zeta_{33}}{\eta_4} \prod^4 (\eta_i N_i)$ | $(\#\mathbb{W}_4^1 + 3\#\mathbb{W}_4^2 + +4\#\mathbb{W}_4^3)$SOS |

***Tabla A.1:*** Núm. de optimizaciones en el algoritmo progresivo.

próximos a 1, sólo el último sumando de $C^2_{LP}$ ya será igual a $C^1_{LP}$. Por tanto, es necesario establecer un criterio que indique qué valor intermedio de $\eta_i$ hace mejor uno u otro algoritmo.
Para ello, se propone un nuevo procedimiento, consistente en aplicar el algoritmo A.1 con $\gamma - 1$ particiones para añadir posteriormente la última partición siguiendo el algoritmo A.2. El número de LPs a resolver en este caso sería:

$$C^3_{LP} = \prod_{i=1}^{\gamma-1} N_i + N_\gamma \prod_{i=1}^{\gamma-1} \eta_i N_i \tag{A.12}$$

Comparando la expresión anterior con $C^1_{LP}$ podemos llegar a conclusiones respecto a $C^2_{LP}$. Si se obtiene que $C^3_{LP} < C^1_{LP}$, se tiene:

$$\begin{aligned} & C^3_{LP} \leq C^1_{LP} \\ & \prod_{i=1}^{\gamma-1} N_i + N_\gamma \prod_{i=1}^{\gamma-1} \eta_i N_i \leq \prod_{i=1}^{\gamma} N_i \end{aligned} \tag{A.13}$$

Es evidente que, si la expresión anterior es válida para un $\gamma$ cualquiera, es posible dar otro paso separando el primer sumando de (A.12) en otros dos:

$$\prod_{i=1}^{\gamma-2} N_i + N_{\gamma-1} \prod_{i=1}^{\gamma-2} \eta_i N_i + N_\gamma \prod_{i=1}^{\gamma-1} \eta_i N_i \leq \prod_{i=1}^{\gamma-1} N_i + N_\gamma \prod_{i=1}^{\gamma-1} \eta_i N_i \leq \prod_{i=1}^{\gamma} N_i$$

Procediendo de igual forma con todos los términos, se llega a $C^2_{LP} \leq C^1_{LP}$.
Por tanto, en aquellos casos en que se demuestre que se cumple (A.13) se probará que el algoritmo progresivo es mejor que el simultáneo. No obstante, si se obtiene que la desigualdad es falsa, no será posible extraer conclusiones respecto al coste de ambos algoritmos. Operando desde (A.13) se obtiene:

$$\begin{aligned} & \prod_{i=1}^{\gamma-1} N_i + N_\gamma \prod_{i=1}^{\gamma-1} \eta_i N_i \leq \prod_{i=1}^{\gamma} N_i \\ & 1 + N_\gamma \prod_{i=1}^{\gamma-1} \eta_i \leq N_\gamma \\ & \prod_{i=1}^{\gamma-1} \eta_i \leq \frac{N_\gamma - 1}{N_\gamma} \end{aligned}$$

Teniendo en cuenta que $\eta_1 = 1$, se define $\eta_m$ como la media geométrica del resto de coeficientes $\eta_i$:

$$(\eta_m)^{\gamma-2} = \prod_{i=2}^{\gamma-1} \eta_i$$

Podemos obtener el valor límite de $\eta_m$ que hace que $C^3_{LP} \leq C^1_{LP}$ como:

$$\eta_m \leq \left( \frac{N_\gamma - 1}{N_\gamma} \right)^{\frac{1}{\gamma-2}} \tag{A.14}$$

La figura A.1 representa dicho valor límite frente al número de particiones, $\gamma$, para diferentes valores de $N_\gamma$. Puede comprobarse que, a medida que aumenta el valor de $\gamma$, el $\eta_m$ límite es cada vez más cercano a 1. Análogamente, a medida que aumenta $N_\gamma$ las curvas se desplazan acercándose también a 1.
Valores de $\eta_m$ límite cercanos a uno significan que, dadas dos particiones, el número de regiones factibles después de realizar todas las intersecciones es muy cercano al producto del número de regiones de las dos particiones, es decir, que casi todas las regiones de una y otra partición pueden intersectarse entre sí. Aunque este hecho podría producirse, en general es poco probable y experimentalmente puede comprobarse que el valor de $\eta_m$ suele encontrarse bastante por debajo del valor mínimo que se aprecia en la figura A.1, 0,75. Teniendo en cuenta además que como ya se ha comentado no se está comparando $C^1_{LP}$ con $C^2_{LP}$ sino con $C^3_{LP}$, que puede estar bastante por encima del anterior, puede afirmarse que, en términos del número de problemas de programación lineal a resolver, el algoritmo progresivo será menos costoso que el simultáneo en la mayoría de las ocasiones.

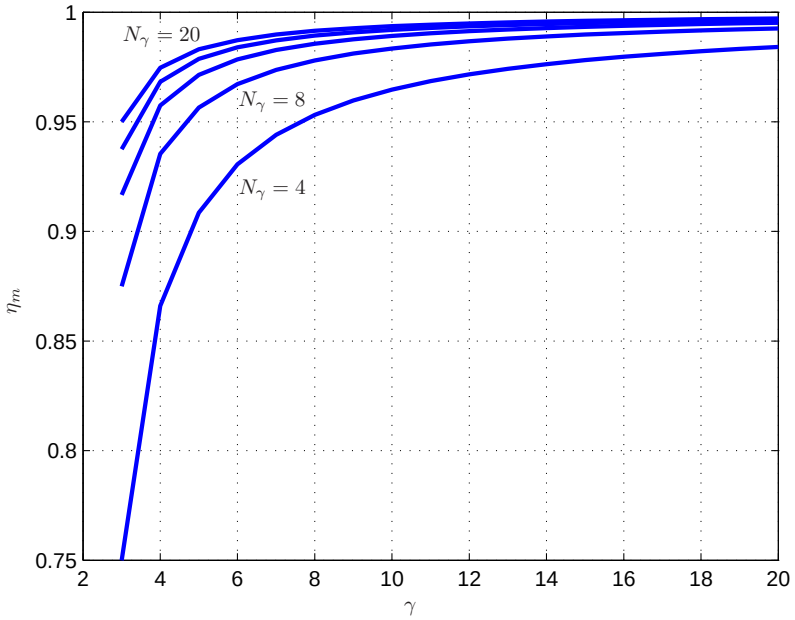


***Figura A.1:*** Valor límite de $\eta_m$.

A continuación, se comparará el número de problemas SOS a resolver mediante ambos algoritmos. En primer lugar, reescribimos (A.2) teniendo en cuenta que el coeficiente que indica las regiones lineales finales factibles, $\eta$ puede escribirse como el producto de los coeficientes que se van obteniendo al añadir

cada partición, los $\eta_i$ definidos para el algoritmo progresivo:

$$C^1_{SOS} = \gamma\eta\prod_{i=1}^{\gamma} N_i = \gamma\prod_{i=1}^{\gamma}(\eta_i N_i) \tag{A.15}$$

Al igual que ocurre con el número de problemas LP a resolver, no existe un mejor algoritmo para absolutamente todas las instancias del problema, como puede deducirse de la observación de los dos costes. En efecto, uno de los casos límites se produce cuando, para todos los $i$ se tiene:

$$\begin{aligned}\frac{\eta_{i(i-1)}\zeta_{(i-1)(i-1)}}{\eta_i} &= 1\\ \frac{\eta_{ij}\zeta_{(i-1)j}}{\eta_i} &= 0 \qquad j = 1\ldots i-2\end{aligned}$$

En este caso, se tiene:

$$C^2_{SOS} = \sum_{i=2}^{\gamma} i\prod_{j=1}^{i}(\eta_j N_j) = C^1_{SOS} + \sum_{i=2}^{\gamma-1} i\prod_{j=1}^{i}(\eta_j N_j) \geq C^1_{SOS}$$

En el otro extremo, se produce un caso límite cuando para todos los $i$ se cumple:

$$\begin{aligned}\frac{\eta_{i1}\zeta_{(i-1)1}}{\eta_i} &= 1\\ \frac{\eta_{ij}\zeta_{(i-1)j}}{\eta_i} &= 0 \qquad j = 2\ldots i-1\end{aligned}$$

En este caso:

$$C^2_{SOS} = \sum_{i=2}^{\gamma}\prod_{j=1}^{i}(\eta_j N_j)$$

Para comparar la ecuación anterior con $C^1_{SOS}$ es útil tener en cuenta que, al añadir una intersección nueva, $\mathbb{X}_i$, e intersectar con la anterior,$\mathbb{T}_{i-1}$, el número de regiones tras dicha intersección siempre va a crecer. Esto es así debido a que cualquier partición abarca todo el espacio de estados. Por tanto, todas las regiones existentes hasta el momento en $\mathbb{T}_{i-1}$ intersectarán al menos con una región de la nueva partición $\mathbb{X}_i$, cuando no con más. Definimos un nuevo parámetro, $\kappa_i$, que representa cuánto crece el número de regiones al añadir sobre el total una nueva partición:

$$\#\mathbb{T}_i = \eta_i N_i \#\mathbb{T}_{i-1} = \kappa_i \#\mathbb{T}_{i-1} \tag{A.16}$$

$$\kappa_i \geq 1 \quad \forall j \tag{A.17}$$

Volviendo a la expresión de coste anterior:

$$C^2_{SOS} = \sum_{i=2}^{\gamma}\prod_{j=1}^{i}(\eta_j N_j) = \sum_{i=2}^{\gamma}\prod_{j=1}^{i}\kappa_j \leq \sum_{i=2}^{\gamma}\prod_{j=1}^{\gamma}\kappa_j = (\gamma-1)\prod_{j=1}^{\gamma}\kappa_j \leq C^1_{SOS}$$

Es evidente que ninguno de los dos algoritmos garantiza menos problemas SOS a resolver para cualquier problema puesto que, dependiendo de las restricciones en cuestión, habrá diferentes números de regiones con una o varias soluciones, y por tanto variará el número de problemas SOS a resolver. No obstante, se pueden suponer diferentes hipótesis y comparar ambos costes en esos casos. Para ello, expresaremos la diferencia entre ambos costes en función del número de regiones de diferente tipo existentes, para facilitar simplificaciones posteriores:

$$C^2_{SOS} - C^1_{SOS} = \sum_{i=2}^{\gamma} \left( \#\mathbb{W}_i + \sum_{j=2}^{i-1} j \#\mathbb{W}_i^j \right) - \gamma \#\mathbb{T}_\gamma \tag{A.18}$$

Se define la función $f(i,j)$:

$$f(i,j) = \frac{\eta_{ij}\zeta_{(i-1)j}}{\eta_i} = \frac{\#\mathbb{W}_i^j}{\#\mathbb{W}_i}$$

Por definición, se tiene:

$$\sum_{j=1}^{i-1} f(i,j) = 1 \tag{A.19}$$

Sustituyendo en (A.18):

$$\begin{aligned} C^2_{SOS} - C^1_{SOS} &= \sum_{i=2}^{\gamma} \left( \#\mathbb{W}_i + \sum_{j=2}^{i-1} j \#\mathbb{W}_i^j \right) - \gamma \#\mathbb{T}_\gamma \\ &= \sum_{i=2}^{\gamma} \left( 1 + \sum_{j=2}^{i-1} j f(i,j) \right) \#\mathbb{W}_i - \gamma \#\mathbb{T}_\gamma \end{aligned}$$

Por otro lado, de (A.16):

$$\#\mathbb{W}_i = \prod_{j=2}^{i} \kappa_j \#\mathbb{W}_1$$

Teniendo en cuenta además que $\#\mathbb{W}_i = \#\mathbb{T}_i$:

$$\begin{aligned} C^2_{SOS} - C^1_{SOS} &= \sum_{i=2}^{\gamma} \left( 1 + \sum_{j=2}^{i-1} j f(i,j) \right) \#\mathbb{W}_i - \gamma \#\mathbb{T}_\gamma \\ &= \sum_{i=2}^{\gamma} \left( 1 + \sum_{j=2}^{i-1} j f(i,j) \right) \prod_{j=2}^{i} \kappa_j \#\mathbb{T}_1 - \gamma \prod_{j=2}^{\gamma} \kappa_j \#\mathbb{T}_1 \\ &= \left( \sum_{i=2}^{\gamma} \frac{\left( 1 + \sum_{j=2}^{i-1} j f(i,j) \right)}{\prod_{j=i+1}^{\gamma} \kappa_j} - \gamma \right) \prod_{j=2}^{\gamma} \kappa_j \#\mathbb{T}_1 \end{aligned}$$

Dado que $\prod_{j=2}^{\gamma} \kappa_j \#\mathbb{T}_1 > 0$, podemos afirmar que $C^2_{SOS} > C^1_{SOS}$ y por tanto el algoritmo simultáneo es mejor que el progresivo siempre que se cumpla:

$$\left( \sum_{i=2}^{\gamma} \frac{\left(1 + \sum_{j=2}^{i-1} j f(i,j)\right)}{\prod_{j=i+1}^{\gamma} \kappa_j} - \gamma \right) > 0$$

Con objeto de simplificar más las expresiones resultantes, se supondrá un valor de $\kappa_j = \kappa$ constante, con lo que:

$$C_h = \sum_{i=2}^{\gamma} \frac{\left(1 + \sum_{j=2}^{i-1} j f(i,j)\right)}{\kappa^{\gamma - i}} - \gamma > 0 \tag{A.20}$$

Una primera hipótesis posible, consiste en asumir que el conjunto de regiones $\mathbb{W}_i$, se divide por igual en los diferentes subconjuntos $\mathbb{W}_i^j$:

$$\frac{\#\mathbb{W}_i^j}{\#\mathbb{W}_i} = f(i,j) = \frac{1}{i-1} \quad j = 1 \ldots i-1$$

Substituyendo $f(i,j)$ en (A.20):

$$C_{h1} = \sum_{i=2}^{\gamma} \frac{\left(1 + \sum_{j=2}^{i-1} \frac{j}{i-1}\right)}{\kappa^{\gamma - i}} - \gamma > 0$$

$$C_{h1} = \sum_{i=2}^{\gamma} \frac{\left(1 + \frac{(i-2)(i+1)}{2(i-1)}\right)}{\kappa^{\gamma - i}} - \gamma > 0$$

$$C_{h1} = \frac{1}{\kappa^{\gamma}} \sum_{i=2}^{\gamma} \left( \frac{i}{2} + \frac{i-2}{i-1} \right) \kappa^i - \gamma > 0 \tag{A.21}$$

La figura A.2 muestra, para diversos valores de $\gamma$, como varía $C_{h1}$ con el parámetro $\kappa$. Puede observarse como, para valores bajos de $\kappa$, $C_{h1} > 0$ y por tanto es menos costoso el algoritmo simultáneo. Esto es razonable y va a ocurrir en todas las hipótesis que se han barajado puesto que, si al añadir una nueva partición el número de regiones no crece mucho, es menos costoso resolver primero los problemas LP y resolver después en cada una de ellas todos los SOS necesarios.

La ecuación (A.21) puede resolverse numéricamente para obtener el valor crítico de $\kappa$ ($\kappa_c$) que hace $C_{h1} = 0$. La figura A.3 representa dicho $\kappa_c$ en función de $\gamma$. Para $\kappa < \kappa_c$, será mejor el algoritmo simultáneo y para $\kappa > \kappa_c$ el progresivo. Se observa como, a partir de un determinado $\gamma$, $\kappa_c$ se estabiliza.

La primera hipótesis supuesta (que hay el mismo número de regiones $\mathbb{W}_i^j$ independientemente del valor de $j$), en muchos casos no es realista puesto que, en general, lo más habitual es que haya una mayoría de regiones con una única solución (tal y como ocurre, por ejemplo, en el ejemplo 4.6). Para reflejar esto,

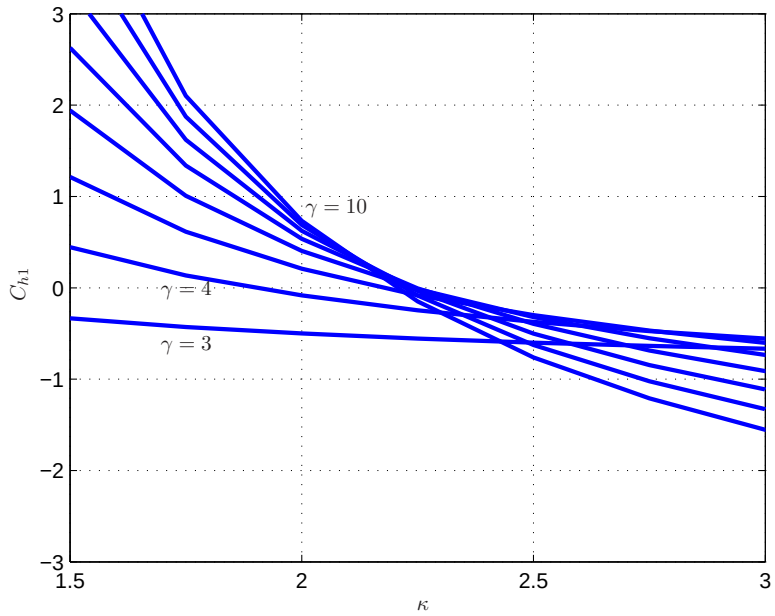


***Figura A.2:*** Comparación de algoritmos. Hipótesis 1.

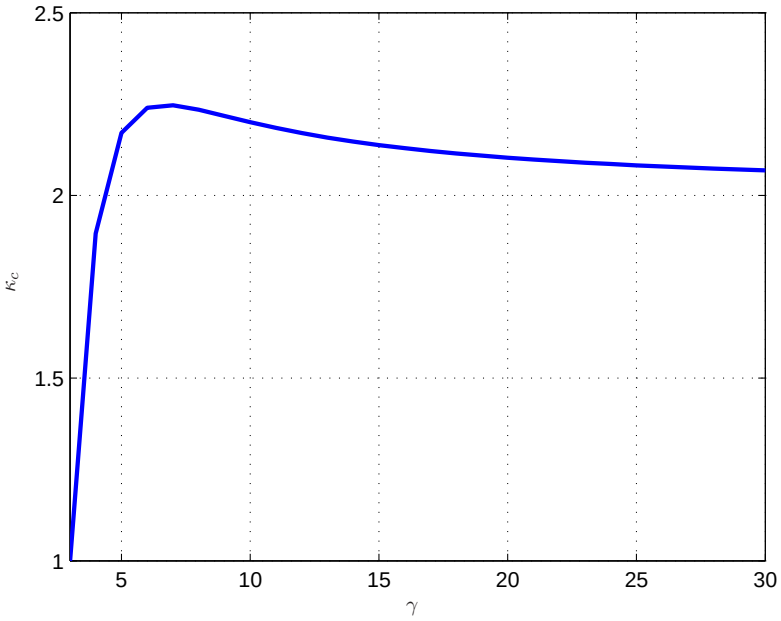


***Figura A.3:*** $\kappa$ crítico. Hipótesis 1.

se introduce una hipótesis diferente en la que hay una fracción de regiones $\mathbb{W}_i^1$ mayor que el resto de $\mathbb{W}_i^j$ (que se reparten equitativamente). Este escenario puede modelarse como:

$$\begin{array}{c} f(i,1) = n \\ f(i,j) = \frac{1-n}{i-2} \quad j = 2 \dots i-1 \end{array}$$

donde $n$ es la fracción de regiones con una única solución.

Podemos comprobar que se satisface (A.19):

$$\sum_{j=1}^{i-1} f(i,j) = n + \sum_{j=2}^{i-1} \frac{1-n}{i-2} = n + 1 - n = 1$$

Substituyendo $f(i,j)$ en (A.20):

$$C_{h2} = \sum_{i=2}^{\gamma} \frac{\left(1 + \sum_{j=2}^{i-1} \frac{j(1-n)}{i-2}\right)}{\kappa^{\gamma-i}} - \gamma > 0$$

$$C_{h2} = \frac{1}{\kappa^{\gamma}} \sum_{i=2}^{\gamma} \left(1 + \sum_{j=2}^{i-1} \frac{j(1-n)}{i-2}\right) \kappa^i - \gamma > 0$$

En la figura A.4 se representan, para diferentes valores de $n$ y $\gamma$ cómo varía $C_{h2}$ con $\kappa$. De nuevo, se calcula numéricamente el valor de $\kappa_c$ (figura A.5) de forma que para $\kappa < \kappa_c$, es mejor el algoritmo simultáneo.

Por último, se analiza una tercera hipótesis, que considere que el número de regiones $\#\mathbb{W}_i^j$ vaya disminuyendo de manera progresiva con $j$. Esto tiene sentido puesto que a medida que aumenta el número de posibles soluciones, es menos probable que se den todas en una misma región lineal. Una posible función $f$ que satisface esta condición es:

$$f(i,j) = \frac{2(i-j)}{i(i-1)}$$

De nuevo, podemos comprobar que se satisface (A.19):

$$\sum_{j=1}^{i-1} f(i,j) = \frac{2}{i(i-1)} \left(\sum_{j=1}^{i-1} i - \sum_{j=1}^{i-1} j\right) = \frac{2}{i(i-1)} \left(i(i-1) - \frac{i(i-1)}{2}\right) = 1$$

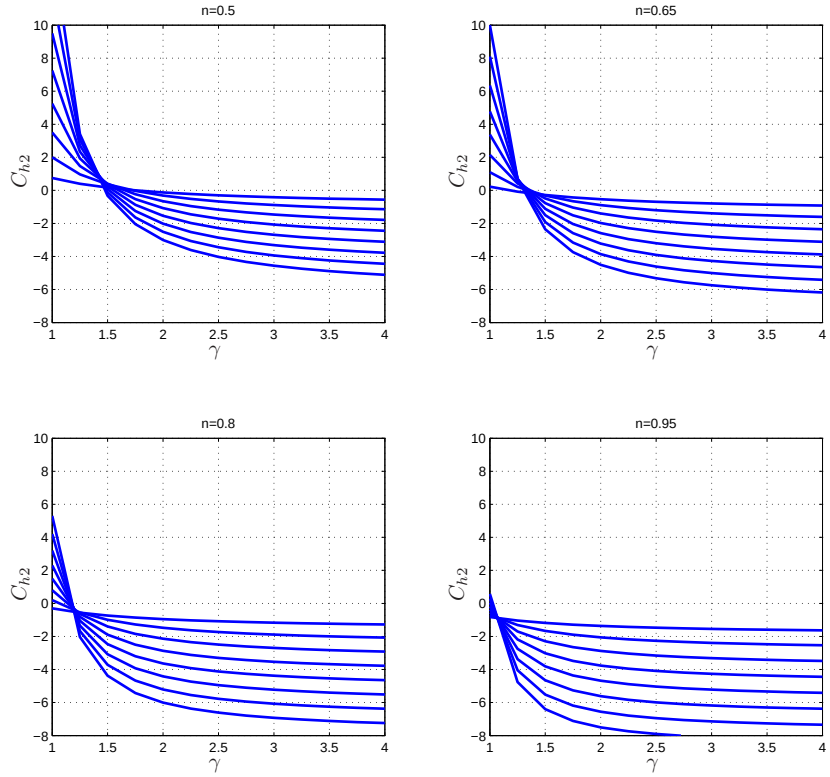


***Figura A.4:*** Comparación de algoritmos. Hipótesis 2.

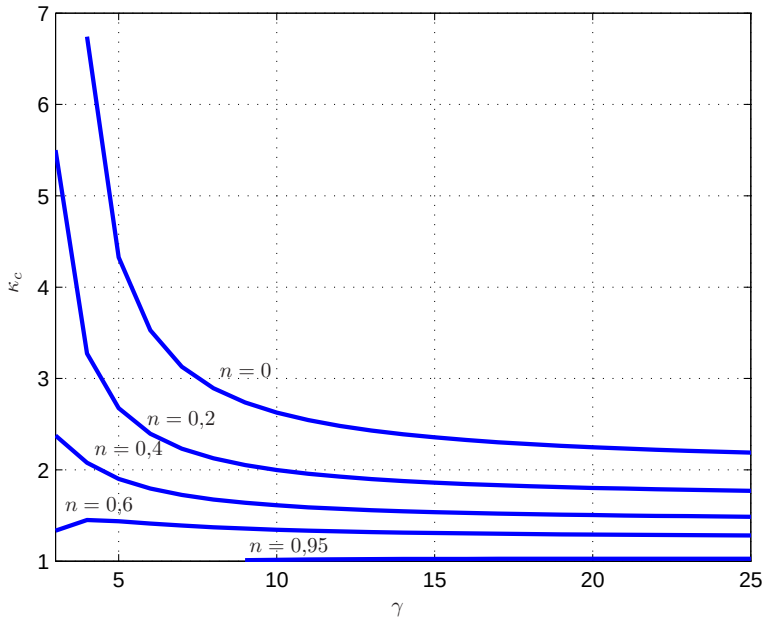


***Figura A.5:*** $\kappa$ crítico. Hipótesis 2.

Substituyendo $f(i,j)$ en (A.20):

$$C_{h3} = \sum_{i=2}^{\gamma} \frac{\left(1 + \sum_{j=2}^{i-1} \frac{2j(i-j)}{i(i-1)}\right)}{\kappa^{\gamma-i}} - \gamma > 0$$

$$\frac{1}{\kappa^{\gamma}} \sum_{i=2}^{\gamma} \left(1 + \frac{(i-2)(i+1)}{i-1} - \frac{2}{i(i-1)} \left(\frac{(i-1)i(2i-1)}{6} - 1\right)\right) \kappa^i - \gamma > 0$$

$$\frac{1}{\kappa^{\gamma}} \sum_{i=2}^{\gamma} (i - 7/2)\, \kappa^i - \gamma > 0$$

En las figuras A.4 y A.5 se representan, respectivamente, $C_{h3}$ frente a $\kappa$ para diferentes $\gamma$ y $\kappa_c$ frente a $\gamma$.

Tras el estudio de las diferentes hipótesis, se llega a las siguientes conclusiones:

- Si se cumple la hipótesis 1, para valores de $\gamma$ bajos el algoritmo simultáneo será el más adecuado sólo si el valor de $\kappa$ es también muy bajo. Para valores de $\gamma$ altos será mejor si $\kappa < 2{,}25$ (figura A.3).
- Si se cumple la hipótesis 2, para $n$ moderadamente altas ($n > 0{,}5$) valores relativamente bajos de $\kappa$ ($\kappa < 1{,}5$) hacen mejor el algoritmo progresivo, independiente de $\gamma$ (figura A.5).

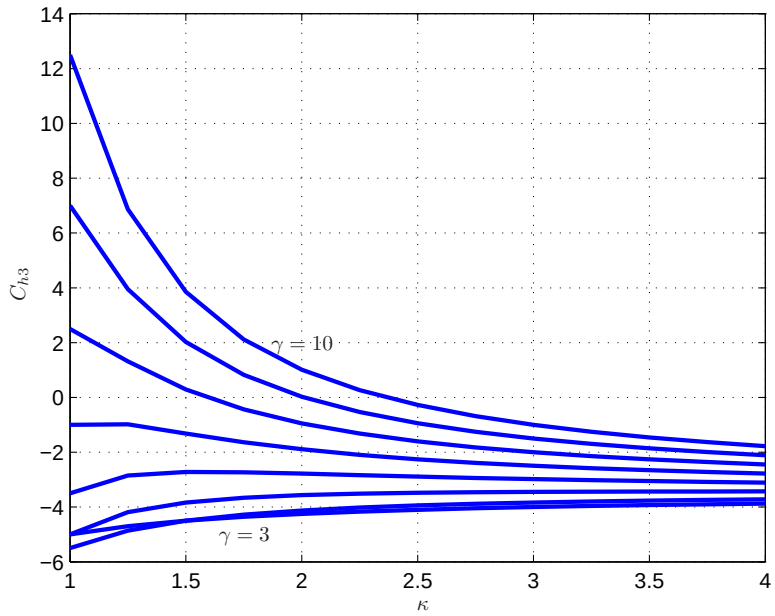


***Figura A.6:*** Comparación de algoritmos. Hipótesis 3.

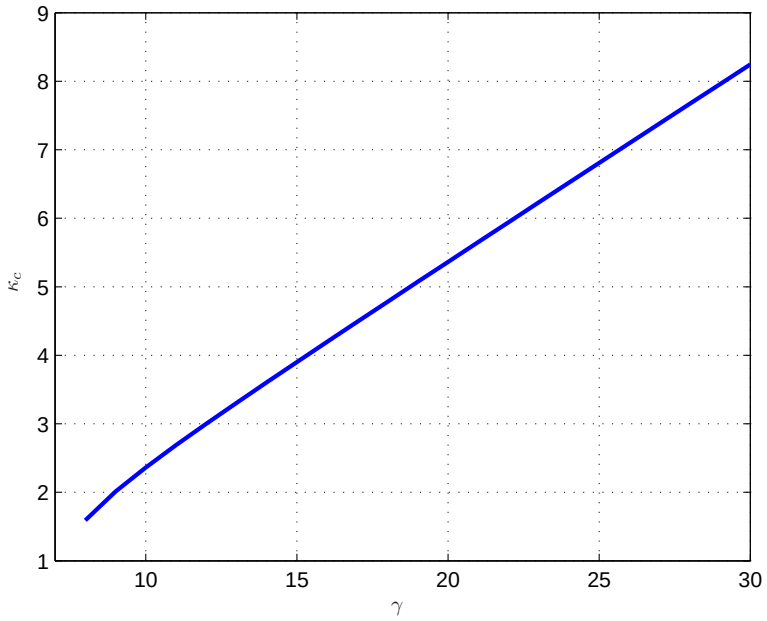


***Figura A.7:*** $\kappa$ crítico. Hipótesis 3.

- Si se cumple la hipótesis 3, para valores bajos de $\gamma$ ($\gamma < 8$) siempre será mejor el algoritmo progresivo, pero a medida que crece $\gamma$, para que siga siéndolo $\kappa$ también debe ser bastante elevado (figura A.7).

Ante la resolución de un problema real, a priori ninguna de las hipótesis propuestas se va a cumplir de manera exacta, puesto que se trata de simplificaciones. No obstante, sí se observa para todas ellas que si el valor de $\gamma$ no es muy elevado, usualmente el algoritmo menos costoso será el progresivo, por lo que será el que se aplique si no se dispone de ninguna información adicional.

# Bibliografía


[AB95] D. Avis and D. Bremner. How good are convex hull algorithms? In *Proceedings of the eleventh annual symposium on Computational geometry*, pages 20–28. ACM New York, NY, USA, 1995.

[AB06] A. Alessio and A. Bemporad. Feasible mode enumeration and cost comparison for explicit quadratic model predictive control of hybrid systems. In *2th IFAC Conf. On Analysis and Design of Hybrid Systems, Alghero, Italy*, 2006.

[ABB+00] J. Abonyi, R. Babuska, M. A. Botto, F. Szeifert, and L. Nagy. Identification and control of nonlinear systems using fuzzy hammerstein models. *Ind.Eng.Chem.Res*, 39(11):4302–4314, 2000.

[ABC98] M. R. Arahal, M. Berenguel, and E. F. Camacho. Neural identification applied to predictive control of a solar plant. *Control Engineering Practice*, 6(3):333–344, 1998.

[AF96] D. Avis and K. Fukuda. Reverse search for enumeration. *Discrete Applied Mathematics*, 65:21–46, 1996.

[ARC05] T. Alamo, D. R. Ramirez, and E. F. Camacho. Efficient implementation of constrained min-max model predictive control with bounded uncertainties: a vertex rejection approach. *Journal of Process Control*, 15(2):149–158, 2005.

[BAG05] M. L. Bergamini, P. Aguirre, and I. Grossmann. Logic-based outer approximation for globally optimal synthesis of process networks. *Computers & Chemical Engineering*, 29(9):1914–1933, 2005.

[Bal88] E. Balas. On the convex hull of the union of certain polyhedra. 1988.

[Bao] M. Baotic. An efficient algorithm for multi-parametric quadratic programming. *ETH Zurich, Institut fur Automatik, Physikstrasse*, 3.

[BBBM01] L. Borrelli, T. Baotic, A. Bemporad, and T. Morari. Efficient on-line computation of constrained optimal control. *Decision and Control, 2001.Proceedings of the 40th IEEE Conference on*, 2, 2001.

[BBBM05] F. Borrelli, M. Baotic, A. Bemporad, and M. Morari. Dynamic programming for constrained optimal control of discrete-time linear hybrid systems. *Automatica*, 41(10):1709–1721, 2005.

[BBKC99] M. A. Botto, T. J. J. V. D. Boom, A. Krijgsman, and J. S. D. Costa. Predictive control based on neural network models with i/o feedback linearization. *International Journal of Control*, 72(17):1538–1554, 1999.

[BBM03a] A. Bemporad, F. Borrelli, and M. Morari. Min-max control of constrained uncertain discrete-time linear systems. *Automatic Control, IEEE Transactions on*, 48(9):1600–1606, 2003.

[BBM03b] F. Borrelli, A. Bemporad, and M. Morari. Geometric algorithm for multiparametric linear programming. *Journal of Optimization Theory and Applications*, 118(3):515–540, 2003.

[BCM94] A. Bemporad, L. Chisci, and E. Mosca. On the stabilizing property of siorhc. *Automatica*, 30(12):2013–2015, 1994.

[BCR98] J. Bochnak, M. Coste, and M. F. Roy. *Real Algebraic Geometry*. Springer, 1998.

[BdBV01] H. H. J. Bloemen, T. J. J. Van den Boom, and H. B. Verbruggen. Model-based predictive control for hammerstein-wiener systems. *International Journal of Control*, 74(5):482–495, 2001.

[BFPS09] F. Borrelli, P. Falcone, J. Pekar, and G. Stewart. Reference governor for constrained piecewise affine systems. *Journal of Process Control*, 19(8):1229–1237, 2009.

[BFT01] A. Bemporad, K. Fukuda, and F. D. Torrisi. Convexity recognition of the union of polyhedra. *Computational Geometry*, 18(3):141–154, 2001.

[BGFB94] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. Linear matrix inequalities in system and control theory. 1994.

[Bie00] L. Biegler. Efficient solution of dynamic optimization and NMPC problems. *Nonlinear model predictive control*, 26, 2000.

[Bla99] F. Blanchini. Set invariance in control. *Automatica*, 35(11):1747–1767, 1999.

[BM99] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35:407–428, 1999.

[BMDP02] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos. The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1):3–20, 1 2002.

[Boo97] T. J. J. Van Den Boom. Robust nonlinear predictive control using feedback linearization and linear matrix inequalities. In *American Control Conference, 1997. Proceedings of the 1997*, volume 5, 1997.

[Bor03] F. Borrelli. *Constrained optimal control of linear and hybrid systems*. Springer Verlag, 2003.

[BR71] D. P. Bertsekas and I. B. Rhodes. On the minimax reachability of target sets and target tubes. *Automatica*, 7:233–247, 1971.

[BSS06] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear programming: theory and algorithms*. John Wiley and Sons, 2006.

[BV04] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[BY09] D. P. Bertsekas and H. Yu. A unifying polyhedral approximation framework for convex optimization. 2009.

[CA98] H. Chen and F. Allgower. Nonlinear model predictive control schemes with guaranteed stability. *Nonlinear Model Based Process Control: Proceedings of the NATO Advanced Study Institute, Antalya, Turkey, August 10-20, 1997*, page 465, 1998.

[CAF03] A. L. Cervantes, O. E. Agamennoni, and J. L. Figueroa. A nonlinear model predictive control system based on wiener piecewise linear models. *Journal of Process Control*, 13(7):655–666, 2003.

[Can03] R. H. Cannon. *Dynamics of physical systems*. Dover Pubns, 2003.

[CB04] E. F. Camacho and C. Bordons. *Model Predictive Control*. Springer, 2004.

[Cha93] B. Chazelle. An optimal convex hull algorithm in any fixed dimension. *Discrete and Computational Geometry*, 10(1):377–409, 1993.

[CM96] D. Chmielewski and V. Manousiouthakis. On constrained infinite-time linear quadratic optimal control. *Systems and Control Letters*, 29(3):121–129, 1996.

[CS82] C. C. Chen and L. Shaw. On receding horizon feedback control. *Control science and technology for the progress of society*, pages 377–382, 1982.

[DBS09] J. Deng, V. M. Becerra, and R. Stobart. Input constraints handling in an mpc/feedback linearization scheme. *International Journal of Applied Mathematics and Computer Science*, 19(2):219–232, 2009.

[DH99] C. E. T. Dórea and J. C. Hennet. (a, b)-invariant polyhedral sets of linear discrete-time systems. *Journal of Optimization Theory and Applications*, 103(3):521–542, 1999.

[dlPnARC07] D. M. de la Peña, T. Alamo, D. R. Ramirez, and E. F. Camacho. Min-max model predictive control as a quadratic program. *Control Theory & Applications, IET*, 1(1):328–333, 2007.

[DMS96] G. Denicolao, L. Magni, and R. Scattolini. On the robustness of receding-horizon control with terminal constraints. *IEEE Transactions on Automatic Control*, 41(3):451–453, 1996.

[DP00] V. Dua and E. N. Pistikopoulos. An algorithm for the solution of multiparametric mixed integer linear programming problems. *Annals of Operations Research*, 99(1):123–139, 2000.

[DPO02] F. J. Doyle, R. K. Pearson, and B. A. Ogunnaike. *Identification and control using Volterra models*. Springer Verlag, 2002.

[FFL01] J. Ferrez, K. Fukuda, and T. M. Liebling. Cuts, zonotopes and arrangements. *Preprint, Swiss Federal Institute of Technology, Lausanne*, 2001.

[FLL01] K. Fukuda, T. M. Liebling, and C. Lütolf. Extended convex hull. *Computational Geometry*, 20(1-2):13–23, 2001.

[FPM97] K. P. Fruzzetti, A. Palazoglu, and K. A. McDonald. Nonlinear model predictive control using hammerstein models. *Journal of Process Control*, 7(1):31–41, 1997.

[Fuk04] K. Fukuda. Frequently asked questions in polyhedral computation. *URL: http://www.ifor.math.ethz.ch/∼fukuda/polyfaq/polyfaq.html*, 2004.

[GBTM04] P. Grieder, F. Borrelli, F. Torrisi, and M. Morari. Computation of the constrained infinite time linear quadratic regulator. *Automatica*, 40(4):701–708, 2004.

[GJB04] J. C. Gomez, A. Jutan, and E. Baeyens. Wiener model identification and predictive control of a ph neutralisation process. *IEE Proceedings-Control Theory and Applications*, 151(3):329–338, 2004.

[GS05] G. C. Goodwin and M. M. Seron. *Constrained Control and Estimation: An Optimisation Approach*. Springer, 2005.

[GT91] E. G. Gilbert and K. T. Tan. Linear systems with state and control constraints: the theory andapplication of maximal output admissible sets. *IEEE Transactions on Automatic Control*, 36(9):1008–1020, 1991.

[GTM08] T. Geyer, F. D. Torrisi, and M. Morari. Optimal complexity reduction of polyhedral piecewise affine systems. *Automatica*, 44(7):1728–1740, 2008.

[GWB90] M. Gevers, V. Wertz, and R. Bitmead. *Adaptive Optimal Control: The Thinking Man's GPC*. Prentice Hall, 1990.

[HA91] E. Hernandez and Y. Arkun. A non-linear DMC controller: Some modeling and robustness considerations. In *Proceedings of the American Control Conference*, pages 2355–2360, 1991.

[HSB01] W. Heemels, B. De Schutter, and A. Bemporad. Equivalence of hybrid dynamical models. *Automatica*, 37(7):1085–1091, 2001.

[Jos02] M. Joswig. Beneath-and-beyond revisited. *Arxiv preprint math.MG/0210133*, 2002.

[Jur06] F. Jurado. Predictive control of solid oxide fuel cells using fuzzy hammerstein models. *Journal of Power Sources*, 158(1):245–253, 2006.

[JZ04] M. Joswig and G. M. Ziegler. Convex hulls, oracles, and homology. *Journal of Symbolic Computation*, 38(4):1247–1259, 2004.

[KCR98] B. Kouvaritakis, M. Cannon, and J. Rossiter. Stability, feasibility, optimality and the number of degrees of freedom in constrained predictive control. 1998.

[KCR99] B. Kouvaritakis, M. Cannon, and J. A. Rossiter. Non-linear model based predictive control. *International Journal of Control*, 72(10):919–928, 1999.

[Ker00] E. C. Kerrigan. Robust constraint satisfaction: Invariant sets and predictive control, 2000.

[Key98] R. De Keyser. A gentle introduction to model based predictive control. In *EC-PADI2 International Conference on Control Engineering and Signal Processing*, pages 18–26, 1998.

[KG88] S. S. Keerthi and E. G. Gilbert. Optimal infinite-horizon feedback laws for a general class of constrained discrete-time systems: Stability and moving-horizon approximations. *Journal of Optimization Theory and Applications*, 57(2):265–293, 1988.

[KH97] M. J. Kurtz and M. A. Henson. Input-output linearizing control of constrained nonlinear processes. *Journal of Process Control*, 7(1):3–18, 1997.

[Kha86] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *The International Journal of Robotics Research*, 5(1):90, 1986.

[Kur05] A. B. Kurzhanski. Dynamic optimization for nonlinear target control synthesis. *Nonlinear Control Systems 2004*, page 21, 2005.

[KY00] J. K. Kuchar and L. C. Yang. A review of conflict detection and resolution modeling methods. *IEEE Transactions on Intelligent Transportation Systems*, 1(4):179–189, 2000.

[LHWB06] M. Lazar, W. Heemels, S. Weiland, and A. Bemporad. Stabilizing model predictive control of hybrid systems. *IEEE Transactions on Automatic Control*, 51(11):1813, 2006.

[Lof04] J. Lofberg. Yalmip: a toolbox for modeling and optimization in matlab. *Computer Aided Control Systems Design, 2004 IEEE International Symposium on*, pages 284–289, 2004.

[LP01] D. Lutterkort and J. Peters. Optimized refinable enclosures of multivariate polynomial pieces. *Computer Aided Geometric Design*, 18(9):851–863, 2001.

[Mea97] E. S. Meadows. Dynamic programming and model predictive control. In *1997 American Control Conference, 15 th, Albuquerque, NM*, pages 1635–1639, 1997.

[MM88] D. Q. Mayne and H. Michalska. Receding horizon control of nonlinear systems. In *Decision and Control, 1988., Proceedings of the 27th IEEE Conference on*, pages 464–465, 1988.

[MM93] H. Michalska and D. Q. Mayne. Robust receding horizon control of constrained nonlinear systems. *IEEE Transactions on Automatic Control*, 38(11):1623–1633, 1993.

[MPJMM09] S. Mahmoodi, J. Poshtan, M. R. Jahed-Motlagh, and A. Montazeri. Nonlinear model predictive control of a ph neutralization process based on wiener-laguerre model. *Chemical Engineering Journal*, 146(3):328–337, 2009.

[MR02] D. Q. Mayne and S. V. Rakovic. Optimal control of constrained piecewise affine discrete time systems using reverse transformation. In *Decision and Control, 2002, Proceedings of the 41st IEEE Conference on*, volume 2, 2002.

[MRRS00] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36:789–814, 2000.

[MS97] L. Magni and R. Sepulchre. *Systems and Control Letters*, 32(4):241–245, 1997.

[NMS96] G. De Nicolao, L. Magni, and R. Scattolini. Stabilizing nonlinear receding horizon control via a nonquadratic penalty. *Proceedings of the IMACS multiconference CESA*, 1:185–187, 1996.

[NPR99] S. J. Norquay, A. Palazoglu, and J. A. Romagnoli. Application of Wiener model predictive control (WMPC) to a pH neutralization experiment. *IEEE Transactions on Control Systems Technology*, 7(4):437–445, 1999.

[Par00] P. A. Parrilo. Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization, 2000.

[Par03] P. A. Parrilo. Semidefinite programming relaxations for semialgebraic problems. *Mathematical Programming*, 96(2):293–320, 2003.

[PBGNS06] E. Pérez, X. Blasco, S. García-Nieto, and J. Sanchis. Diesel engine identification and predictive control using wiener and hammerstein models. In *2006 IEEE Computer Aided Control System Design (CACSD), International Conference on Control Applications (CCA) and International Symposium on Intelligent Control (ISIC)*, pages 2417–2423, 2006.

[PCPC05] M. Peña, E. F. Camacho, S. Piñón, and R. Carelli. Model predictive controller for piecewise affine system. In *Proceedings of the 16th IFAC World Congress*, 2005.

[PLS97] R. S. Patwardhan, S. Lakshminarayanan, and S. L. Shah. Nonlinear model predictive control using PLS based Hammerstein and Wiener models. In *AI Ch. E. Meeting, LA, Session*, volume 217, 1997.

[PLS98] R. S. Patwardhan, S. Lakshminarayanan, and S. L. Shah. Constrained nonlinear MPC using Hammerstein and Wiener models: PLS framework. *AICHE Journal*, 44(7):1611–1622, 1998.

[PN97] J. A. Primbs and V. Nevistic. Constrained finite receding horizon linear quadratic control. In *IEEE Conference on Decision and Control*, volume 4, pages 3196–3201. IEEE, 1997.

[RC06] D. R. Ramirez and E. F. Camacho. Piecewise affinity of min-max MPC with bounded additive uncertainties and a quadratic criterion. *Automatica*, 42(2):295–302, 2006.

[RGK+04] S. V. Rakovic, P. Grieder, M. Kvasnica, D. Q. Mayne, and M. Morari. Computation of invariant sets for piecewise affine discrete time systems subject to bounded disturbances. In *IEEE Conference on Decision and Control*, pages 1418–1423, 2004.

[RM93] J. B. Rawlings and K. R. Muske. The stability of constrained receding horizon control. *IEEE Transactions on Automatic Control*, 38(10):1512–1516, 1993.

[RM05] S. V. Rakovic and D. Q. Mayne. Robust time optimal obstacle avoidance problem for constrained discrete time systems. In *IEEE Conference on Decision and Control*, volume 44, page 981. IEEE; 1998, 2005.

[RM07] S. V. Rakovic and D. Q. Mayne. Robust model predictive control for obstacle avoidance: discrete time case. *Lecture Notes in Control and Information Sciences*, 358:617, 2007.

[Ros03] J. A. Rossiter. *Model-Based Predictive Control: A Practical Approach*. CRC Press, 2003.

[SGD03] M. M. Seron, G. C. Goodwin, and J. A. De Dona. Characterisation of receding horizon control for constrained linear systems. *Asian Journal of Control*, 5(2):271–286, 2003.

[SKJ+06] J. Spjøtvold, E. C. Kerrigan, C. N. Jones, P. Tøndel, and T. A. Johansen. On the facet-to-facet property of solutions to convex parametric quadratic programs. *Automatica*, 42(12):2209–2214, 2006.

[SL91] J. J. E. Slotine and W. Li. *Applied nonlinear control*. Prentice-Hall Englewood Cliffs, NJ, 1991.

[SMR99] P. O. M. Scokaert, D. Q. Mayne, and J. B. Rawlings. Suboptimal model predictive control (feasibility implies stability). *IEEE Transactions on Automatic Control*, 44(3):648–654, 1999.

[SP99] E. M. B. Smith and C. C. Pantelides. A symbolic reformulation/spatial branch-and-bound algorithm for the global optimisation of nonconvex minlps. *Computers & Chemical Engineering*, 23(4-5):457–478, 1999.

[SR98] P. O. M. Scokaert and J. B. Rawlings. Constrained linear quadratic regulation. *IEEE Transactions on Automatic Control*, 43(8):1163–1169, 1998.

[Ste73] G. Stengle. A Nullstellensatz and a Positivstellensatz in semialgebraic geometry. *Mathematische Annalen*, 207(2):87–97, 1973.

[Stu99] J. F. Sturm. Using SeDuMi 1.02, a Matlab toolbox for optimization over symmetric cones. *Optimization Methods and Software*, 11(1):625–653, 1999.

[TI01] S. Townsend and G. W. Irwin. Nonlinear model based predictive control using multiple local models. *Nonlinear predictive control: theory and practice*, 2001.

[Tiw07] H. R. Tiwary. On the hardness of minkowski addition and related operations. In *Proceedings of the twenty-third annual symposium on Computational geometry*, pages 306–309. ACM Press New York, NY, USA, 2007.

[TJB03a] P. Tøndel, T. A. Johansen, and A. Bemporad. An algorithm for multi-parametric quadratic programming and explicit MPC solutions. *Automatica*, 39(3):489–497, 2003.

[TJB03b] P. Tøndel, T. A. Johansen, and A. Bemporad. Evaluation of piecewise affine control via binary search tree. *Automatica*, 39(5):945–950, 5 2003.

[TO95] O. Toker and H. Ozbay. On the np-hardness of solving bilinear matrix inequalities andsimultaneous stabilization with static output feedback. In *American Control Conference, 1995. Proceedings of the*, volume 4, 1995.

[VB96] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, 1996.

[VB00] J. G. VanAntwerp and R. D. Braatz. A tutorial on linear and bilinear matrix inequalities. *Journal of Process Control*, 10(4):363–385, 2000.

[vNMS+98] M. van Nieuwstadt, P. E. Moraal, I. V. K. A. Stefanopoulou, P. Wood, and M. Criddle. Decentralized and multivariable designs for egr-vgt control of a diesel engine. In *Advances in automotive control 1998: a proceedings volume from the 2nd IFAC workshop, Mohican State Park, Loudonville, Ohio, USA, 26 February-1 March 1998*, page 189. Pergamon Pr, 1998.

[ZZ01] A. Zheng and W. Zhang. Non-linear predictive control, chapter: Computationally efficient non-linear model predictive control algorithm for control of constrained non-linear systems. *IEE Control Engineering series*, 2001.