UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería del Diseño

DISEÑO DE CONTROLES INVARIANTES USANDO ÁLGEBRA DIFERENCIAL

MEMORIA PRESENTADA POR:

Emilio Ruiz Moreno

Máster Universitario en Ingeniería Mecatrónica

DIRECTOR:

Ramón Manuel Blasco Giménez

CODIRECTOR:

Enric Picó Marco

JULIO 2018

Agradecimientos a Yadira Boada y Alejandro Vignoni por su ayuda y consejo en la parte bioquímica de este proyecto.

Índice

Índice de figuras

Índice de tablas

1. Introducción

La teoría de sistemas de control se ocupa del análisis y el diseño de componentes que actúen sobre la dinámica de un sistema objeto en una configuración que imite el comportamiento de un sistema dinámico objetivo al que llamamos modelo de referencia o simplemente referencia cuando se trata de un valor fijo o una trayectoria a seguir por una determinada variable. Esta teoría se basa en la información y su conocimiento, acerca del comportamiento de los sistemas, que queda reflejada en las variables que influyen en su dinámica y en las ecuaciones que conforman.

La teoría de sistemas de control se aplica en diversas disciplinas desde la economía hasta la biología pasando por todo tipo de procesos industriales. En este proyecto nos centramos en los sistemas que estudia la ingeniería de control y por ende la mecatrónica, si la consideramos como una disciplina que integra la ingeniería mecánica, electrónica, informática y automática.

En general, el punto de partida para el análisis de un sistema de control, es su representación a través de un modelo matemático que recoja sus aspectos de interés, generalmente como un conjunto de ecuaciones diferenciales y/o algebraicas. La mayoría de los modelos matemáticos usados tradicionalmente son lineales. De hecho, los modelos lineales son, usualmente, más manejables que los no lineales, y pueden representar de forma precisa el comportamiento de sistemas reales en muchos casos útiles. Sin embargo, los avances tecnológicos han generado una enorme variedad de nuevos problemas y aplicaciones que son, en esencia, no lineales. Por ejemplo, fenómenos no lineales se observan comúnmente en aplicaciones modernas de ingeniería y entre ellas, destacamos por su papel en este proyecto, los sistemas bioquímicos que, por lo general, no son lineales.

En ocasiones, esta fenomenología no lineal, no puede aproximarse adecuadamente a través de la dinámica de modelos lineales. Razón ineludible para el uso de modelos no lineales y el desarrollo de conceptos y herramientas de sistemas no lineales de control, que si bien no existe un método genérico como en el caso de sistemas lineales, si lo hay para conjuntos de sistemas que compartan ciertas características. Sin ir más lejos, en este proyecto estudiamos un método de diseño de control aplicable a sistemas no lineales descritos mediante ecuaciones diferenciales polinomiales y/o algebraicas.

Pero antes de abordar el método de diseño, conviene recordar que en este proyecto, como su título indica, nos centramos en los controles invariantes, por lo que es necesario, y de hecho hacemos en el siguiente apartado, responder a la pregunta ¿qué son y cuál es su área de aplicación?.

1.1. Control invariante

Conceptualmente un control invariante es aquel que solo funciona al actuar sobre la variedad, valga la redundancia, invariante, que contiene todas las soluciones del sistema dinámico de referencia, asegurándonos por definición que cualquier trayectoria dentro de la variedad permanece dentro de la variedad. Es decir, una vez que el control es efectivo lo es para toda la evolución dinámica del sistema, suponiendo que no hay perturbaciones y que el modelo es ideal.

Con el siguiente ejemplo pretendemos ilustrar, de forma práctica, conceptos que ayuden a interiorizar el significado de control invariante.

Ejemplo: Sea el sistema de ecuaciones diferenciales, en coordenadas polares, del movi-

miento de una partícula dado por:

$$\begin{aligned} \dot{r} &= r(1-r) \\ \dot{\theta} &= 1 \end{aligned} \tag{1}$$

Analíticamente, a partir del estudio de sus soluciones, podemos concluir que tanto la órbita circular para $r = 1$ como el origen, punto crítico en $r = 0$, son conjuntos invariantes ya que el sistema siempre permanece en ellos puesto que $\dot{r} = 0$. Gráficamente se observa mejor este hecho.

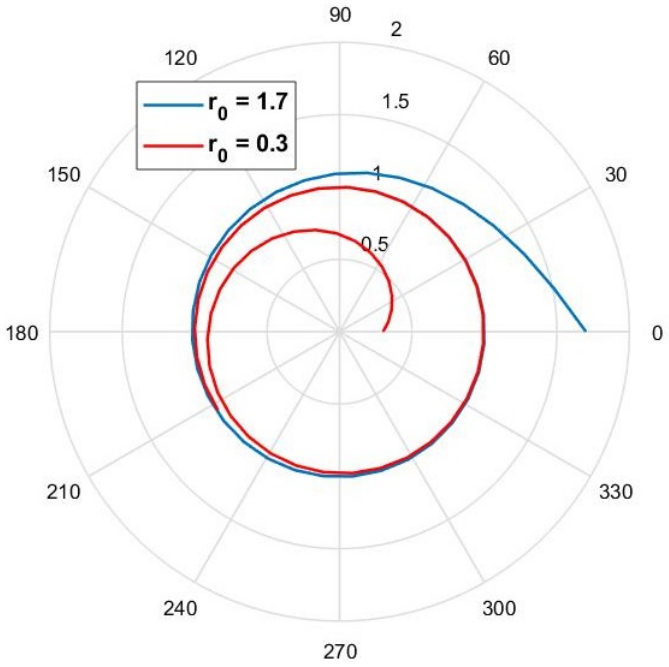


Figura 1: Trayecorias de la partícula según condición inicial.

En primer lugar, como podemos observar en la figura (1), las condiciones iniciales determinan el punto de partida de la trayectoria. En segundo lugar, cuando una trayectoria alcanza la órbita $r = 1$, permanece en ella, como suponíamos al tratarse de una variedad invariante y tercero, parece que las órbitas independientemente del punto de origen, excepto para el punto crítico $r = 0$, acaban estabilizándose en la órbita circular $r = 0$.

Este último punto, el tercero, es lo que se conoce como variedad atractiva. Por ejemplo, cualquier corona circular de radios a y b con $a > 1$ y $b < 1$ es atractiva ya que:

$$\begin{aligned} r = a > 1 \rightarrow \dot{r} < 0 \\ r = b < 1 \rightarrow \dot{r} > 0 \end{aligned}$$

cualquier trayectoria que empiece en la corona se desarrolla hasta alcanzar la órbita circular $r = 1$.

La variedad conformada por el conjunto de las soluciones del modelo de referencia, no tiene porque ser atractiva. En el caso de no serlo, o bien escogemos adecuadamente las condiciones

iniciales para que el control actúe directamente sobre la variedad invariante (volviendo al símil del ejemplo sería empezar directamente en $r = 1$) o diseñamos una segunda capa de control que guíe al sistema hasta la variedad invariante (haciendo desde el punto de vista del control la variedad atractiva).

En el proyecto solo nos centramos en la parte de diseño del control invariante, dejando la segunda capa de control atractiva, en el caso de que fuera necesaria, para futuros estudios.

1.2. Objetivos

El objetivo principal de este proyecto es el estudio de un método de diseño de controles invariantes. Este método emplea técnicas de análisis computacional basadas en álgebra diferencial que, de forma breve, es el área del álgebra que estudia las ecuaciones diferenciales. Estas técnicas necesitan que la dinámica, tanto del sistema objeto como del objetivo, esté expresada a través de ecuaciones diferenciales polinomiales y/o algebraicas.

Es la restricción, en la forma del modelo matemático, impuesta por la técnica computacional estudiada, la que nos define el conjunto de sistemas dinámicos a los que se dirige este proyecto. Entre los sistemas candidatos a estudio, nos centramos en un sistema bioquímico no lineal, cuyo modelo matemático es expresable a partir de ecuaciones diferenciales polinomiales y/o algebraicas. En él analizamos todo el proceso de diseño, desde la creación del modelo matemático hasta la implementación del control invariante.

En definitiva y de forma secuencial, a lo largo del proyecto estudiaremos las características y los fundamentos matemáticos de la técnica computacional de diseño de control empleada. A partir de este punto entenderemos las restricciones que impone y por que el control resultante es invariante y finalmente, analizaremos el método tanto teórica como experimentalmente aplicándolo a un sistema bioquímico no lineal.

1.3. Relevancia del tema ¿aportaciones?

El tema del proyecto gira en torno al diseño de controles invariantes. Para ello nos apoyamos en técnicas computacionales basadas en álgebra diferencial. Estas técnicas están implementadas, entre otros, por un programa de cálculo simbólico llamado *Maple*, que además es el que empleamos a lo largo del proyecto. Al ser un *software* de uso relativamente normalizado, cualquier usuario puede emplear estas técnicas de forma accesible.

Recordamos que el único requisito del sistema no lineal sobre el que queremos diseñar un control, es que su dinámica sea expresable a partir de ecuaciones diferenciales polinomiales y/o algebraicas. Si esto es así, es posible emplear el método de diseño, aunque los resultados dependerán, obviamente, tanto del propio sistema como de la dinámica objetivo impuesta.

Dentro de los sistemas candidatos a los que podemos aplicar el método de diseño, resaltamos por centrarnos en ellos durante el proyecto, los circuitos genéticos [14]. Éstos, pertenecen al campo de la biología sintética que ha crecido considerablemente durante los últimos quince años. La biología sintética representa una nueva estrategia que promete ampliar al conocimiento básico de la biología así como la creación de sistemas novedosos con aplicaciones prácticas. El diseño de un circuito genético, en ocasiones, es comparado a la programación, donde la célula juega el papel de computador y los circuitos genéticos, en su interior, el papel de programas. Desde esta perspectiva, construir un circuito genético es como introducir un pequeño *script* en un sistema operativo.

En el interior de los circuitos genéticos ocurren reacciones químicas y bioquímicas que pueden ser modeladas a través de ecuaciones, derivadas de la ley cinética de acción de masas, de la forma:

$$\dot{x} = Sv(k, x) \tag{2}$$

donde x es el vector de concentraciones, S es la matriz estequiométrica que determina las relaciones cuantitativas entre los reactivos y productos en el transcurso de una reacción química y v es el vector que, a groso modo, determina a que velocidad se convierten los reactivos en productos, formado por ecuaciones polinomiales multivariable de coeficientes k reales. Que efectivamente corresponde a ecuaciones diferenciales polinomiales.

En resumen, aunque en este proyecto se hace un primer enfoque, la técnica empleada con su debido estudio, posee características que le auguran un largo recorrido dentro del campo de la biología sintética. En concreto, en el diseño de circuitos genéticos y todo ello sin olvidarnos de su aplicabilidad en el resto de campos que trabajen con sistemas no lineales y expresables por ecuaciones diferenciales polinomiales y/o algebraicas.

2. Contextualización

En esta sección vamos a situar y desarrollar algunos aspectos del área de estudio en la que se centra el proyecto. Dividida en dos bloques diferenciados, uno relacionado con sistemas de control y el otro a procesos bioquímicos. Dentro de cada uno de ellos, hablaremos sobre los trabajos previos que han servido como apoyo, desarrollaremos la técnica computacional y el bagaje matemático que nos permite diseñar controles invariantes y describiremos el modelo bioquímico a controlar, que anunciamos se trata un circuito genético. Es decir, un sistema biológico que produce una proteína de interés de acuerdo a las reacciones químicas que se producen dentro de una célula.

2.1. Trabajos previos relacionados

Como hemos mencionado previamente, tanto el método de diseño de control como el modelo genético sobre el que actuamos, se apoyan en trabajos previos.

El proyecto hereda las técnicas computacionales basadas en álgebra diferencial descritas en el artículo *Differential Algebra for Control Systems Design* [11] e ideas del codirector del proyecto, sobre el diseño de controles invariantes usando las técnicas mencionadas. En cuanto al circuito genético el modelo matemático empleado surge de la formulación descrita en el artículo *Resource Competition Shapes the Response of Genetic Circuits* [12] que describe la distribución de recursos en el circuito genético mediante ecuaciones diferenciales polinomiales.

De forma resumida y para agilizar la lectura, empleando terminología que explicamos y/o desarrollamos en los siguientes apartados, describimos los aspectos de interés para el proyecto, en cada uno de los artículos citados.

Empezamos por el artículo titulado *Differential Algebra for Control Systems Design* [11]. Éste comienza enumerando los principales objetivos del álgebra diferencial (DALG por sus siglas en inglés) que son: estudiar, estimar y describir la solución de un sistema de ecuaciones diferenciales polinomial.

En muchos casos puede ser imposible estimar simbólicamente las soluciones, o estas soluciones ser difíciles de manejar. Aun así es útil ser capaces de estudiar y de forma estructurada describir las soluciones ya que a menudo, entender las propiedades del espacio de soluciones y consecuentemente de las soluciones es todo lo necesario para el análisis y diseño del control.

Existen diseños de control sistemáticos para sistemas no lineales en una forma determinada tales como, el bloque de prealimentación (*block feedforward*, BFD en inglés) y la forma de control generalizado (*Generalized control*, GECO en inglés). Varias técnicas de diseño de control han sido desarrolladas basadas en estas formas canónicas, o al menos, casos particulares de estas formas. Para que sean plenamente sistemáticas debe de haber una forma equivalente de transformar un sistema genérico, si es posible, en una de estas formas canónicas.

El artículo presenta una introducción al DALG con el principal foco puesto sobre el algoritmo de *Rosenfeld-Groebner*, la forma BFD y la forma GECO.

Una herramienta básica de la DALG es el algoritmo de *Rosenfeld-Groebner*. Si bien un entendimiento profundo de las técnicas que fundamentan el algoritmo están fuera del campo de estudio del proyecto, por el fuerte trasfondo matemático en álgebra abstracta necesario, el objetivo del algoritmo es reescribir un sistema de ecuaciones en una forma equivalente que sea más fácil de usar para un determinado propósito.

Las entradas del algoritmo de *Rosenfeld-Groebner* son: 1) el sistema que queremos expresar de forma canónica, 2) un orden, al que llamamos *ranking*, entre las variables del sistema. La salida es una familia de sistemas que tienen el mismo conjunto de soluciones que el sistema original.

Todos los sistemas en la descomposición tienen una estructura especial y se dice que son regulares. Si el *ranking* es elegido de forma adecuada, estos sistemas regulares se encuentran en la forma canónica deseada.

Un concepto clave en el proceso de obtener la descomposición de un sistema en sistemas regulares es la reducción, que es una extensión del DALG para el algoritmo de división usual teniendo en cuenta la presencia de derivadas. Para definir un algoritmo de división para polinomios de más de una variable en un anillo polinomial, debemos ser capaces de comparar cada par de monomios y determinar cual es "mayor".

La idea que hay muchos sistemas con las mismas soluciones es tratada asociando un ideal a un sistema de ecuaciones dado. Intuitivamente un ideal I asociado a un conjunto de ecuaciones es un conjunto de infinitas ecuaciones tales que todas se vuelven idénticamente cero cuando las soluciones del sistema original son insertadas. El término diferencial hace referencia al hecho de que la operación derivada debe ser considerada por tratar con ecuaciones diferenciales. El citado conjunto de infinitas ecuaciones está formado por combinaciones lineales de las ecuaciones originales y sus derivadas donde los coeficientes son ecuaciones diferenciales polinomiales genéricas. Finalmente, el término radical indica que ninguna ecuación que se anule con las soluciones del conjunto original queda fuera. En cierto sentido, el radical de un ideal diferencial contiene todos, y solo, los sistemas equivalentes al problema bajo consideración, por lo que todas las posibilidades están contempladas. El algoritmo de *Rosenfeld-Groebner* crea un representación del correspondiente radical, denotado como $\sqrt{I}$ y toma el *ranking* definido por el usuario para generar la descomposición en sistemas regulares.

La principal característica de un sistema regular es que es diferencialmente triangular, es decir ningún elemento pertenece al conjunto de los coeficientes y cualquier elemento se deduce con respecto a los otros.

A continuación vamos a mostrar un ejemplo del artículo en el que se describe la aplicación del algoritmo de *Rosenfeld-Groebner* para la forma canónica BFD. Nótese que el término empleado es reescribir y no reformular ya que las variables siempre mantienen su significado original. Las expresiones obtenidas dependen del *ranking* escogido, algunas serán más sencillas que otras, pero todas ellas estarán en forma BFD.

Formalmente, un sistema en forma BFD se define por r bloques tal que:

$$\begin{aligned} \dot{x}_1 &= f_1(x_1, u) \\ \dot{x}_2 &= f_2(x_1, x_2, u) \\ &\vdots \\ \dot{x}_r &= f_r(x_1, x_2, \ldots, u) \end{aligned} \tag{3}$$

donde $x_i \in \mathbb{R}^{n_i}$ y u es el vector de entradas. Remarcar que un sistema en la forma BFD no es producido directamente, si no que es una colección de polinomios con la forma:

$$P(x_i^{(\alpha)}, \ldots, \dot{x}_i, x_i, \ldots, x_j^{(\beta)}, \ldots, \dot{x}_j, x_j, \ldots, u^{(\sigma)}, \ldots, \dot{u}, u) = 0 \tag{4}$$

que pueden ser transformados a la forma (3) creando nuevas variables auxiliares asociadas a las derivadas, de forma que P sea un polinomio dependiente de las nuevas variables más una cadena de integradores.

Ejemplo: considere el sistema:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_3 &= x_1 \\ x_4 &= x_1 \\ x_4 &= -x_2 - x_3 + u \end{aligned} \tag{5}$$

Usando el algoritmo de *Rosenfeld-Groebner* con el *ranking* de eliminación $[x_1, x_2, x_3, x_4, u]$, siendo el estado x_1 y sus derivadas el más importante frente a la entrada u y sus derivadas que es la menos importante, obtenemos el sistema regular:

$$\begin{aligned} x_1 &= x_4 \\ x_2 &= \dot{x}_4 \\ x_3 &= -\dot{x}_4 - x_4 + u \\ \ddot{x}_4 &= -\dot{x}_4 - x_4 + \dot{u} \end{aligned} \tag{6}$$

y después de reordenar y renombrar las variables como: $z_1 = x_1$, $z_2 = x_2$, $z_3 = x_3$, $z_4 = x_4$, $z_5 = \dot{x}_4$, $z_6 = u$, obtenemos el sistema equivalente:

$$\begin{aligned} z_1 &= z_4 \\ z_2 &= z_5 \\ z_3 &= -z_4 - z_5 + z_6 \\ \dot{z}_4 &= z_5 \\ \dot{z}_5 &= -z_4 - z_5 + v \\ \dot{z}_6 &= v \end{aligned} \tag{7}$$

El precio a pagar por reescribir el sistema en esta forma es que, debido a la presencia de una derivada de la entrada, hay un incremento de uno en el número de estados. Sin embargo, esta situación puede evitarse aplicando el algoritmo de *Rosenfeld-Groebner* con un *ranking* distinto: $[x_4, x_1, x_2, x_3, u]$, sobre el sistema original (5), queda el sistema regular:

$$\begin{aligned} x_4 &= x_3 \\ x_1 &= \dot{x}_3 \\ x_2 &= -\dot{x}_3 - x_3 + u \\ \ddot{x}_3 &= -\dot{x}_3 - x_3 + u \end{aligned} \tag{8}$$

que no tiene derivadas de u. Cuando el sistema (8) de ecuaciones es reescrito en la forma BFD, la descripción tiene solo dos estados como en el sistema original (5) y todos ellos son funciones explicitas de u y de los otros estados.

Debido a que la forma canónica GECO es un caso particular de la forma BFD, se emplea un procedimiento muy similar al ya usado. La principal diferencia es que es necesario encontrar una salida específica dependiente de los estados, las entradas y un número finito de sus derivadas.

En conclusión, este artículo proporciona una introducción al álgebra diferencial y demuestra sus aplicaciones para calcular dos formas canónicas específicas que pueden ser usadas posterior-

mente para el diseño de controladores no lineales. Estos métodos son válidos para cualquier sistema definido por polinomios diferenciales, incluso de forma implícita. Este enfoque tiene ciertas ventajas al transformar el sistema con un cambio de coordenadas. Primero, el tratamiento mediante álgebra diferencial es realizable. Segundo, se conserva las variables originales con su significado físico y finalmente, los sistemas implícitos son tratados del mismo modo que los explícitos.

La idea de aplicar las técnicas del artículo anterior [11], en el diseño de controles invariantes, da lugar al núcleo de este proyecto. Desarrollamos esta idea.

En diferentes marcos de diseño de controladores, el proceso de diseño se divide en dos pasos. Primero, un controlador se diseña para hacer invariante una variedad dada que puede venir definida implícitamente por alguna dinámica objetivo. Segundo, otra capa de control se diseña para hacer que la variedad antes mencionada sea atractiva. Nos queremos centrar en el primer paso, tomando la dinámica con parámetros nominales y definir constructivamente un controlador tal que que las trayectorias del sistema de bucle cerrado cumplan con una restricción dada. Esta puede darse en la forma de una primera integral deseada o como un sistema dinámico que puede, por ejemplo, representar la evolución del error. Los controladores pueden ser estáticos o dinámicos y el método debería poder aplicase a sistemas implícitos. Los sistemas, sin embargo, deben definirse por polinomios diferenciales y así permitir el uso de los métodos computacionales del álgebra diferencial.

Formalmente, nuestra meta es diseñar un controlador con la forma genérica:

$$K(u^{\alpha}, u^{\alpha-1}, \ldots, u, x^{\beta}, x^{\beta-1}, \ldots, x) = 0 \tag{9}$$

tal que el comportamiento en bucle cerrado sea compatible con las restricciones de diseño dadas por un conjunto de ecuaciones diferenciales y/o algebraicas. Debido a características del álgebra diferencial, estas restricciones pueden restringir el conjunto de posibles condiciones iniciales.

Además, buscamos que el método sea constructivo, es decir a partir de un conjunto inicial 'construir' un conjunto nuevo, de ahí que empleemos los algoritmos basados en álgebra diferencial ya vistos. Muchos de ellos ya han sido implementados en *software* de álgebra computacional como *Maple*. Como resultado tratamos con sistemas definidos por polinomios diferenciales multivariable y sus respectivas derivadas. Esto incluye sistemas de control explícitos e implícitos definidos por ecuaciones diferenciales ordinarias incluyendo restricciones algebraicas.

En este contexto, el algoritmo de *Rosenfeld-Groebner* se usa para obtener un control del sistema de ecuaciones junto con el modelo de las restricciones deseadas para la dinámica en bucle cerrado. Añadiendo un número suficiente de restricciones obtenemos un sistema (sobre)determinado que contiene implícitamente una expresión para la acción de control u o para una de sus derivadas. Empleamos el hecho de que la salida del algoritmo es diferencialmente triangular para calcular la expresión explícita. Además la elección del tipo de *ranking* permite buscar por expresiones en términos de determinadas variables, por ejemplo las que son mensurables. Una vez ejecutado, debemos comprobar si el sistema en bucle cerrado realmente cumple las restricciones impuestas. Para ello realizamos una prueba de pertenencia que comprueba que la dinámica de referencia pertenece al ideal generado por los polinomios de la dinámica en bucle cerrado y puesto que la prueba solo puede ser llevada a cabo mediante la caracterización de un ideal, nos vemos forzados en muchas ocasiones a aplicar el algoritmo una segunda vez.

Ilustremos los últimos párrafos con un ejemplo secuencial:

1. Tomamos el modelo del proceso:

$$\begin{aligned} \dot{x}_1 &= x_1 + x_1^3 + x_2 \\ \dot{x}_2 &= u \\ y &= x_1^3 + x_2 \end{aligned} \tag{10}$$

2. Codificamos las especificaciones en un conjunto de polinomios diferenciales,

$$\dot{y} = -ky \tag{11}$$

donde la constante k es tratada como un nuevo número usando una extensión del cuerpo.

3. Tomámos tanto el proceso como las especificaciones y aplicamos el algoritmo de *Rosenfeld-Groebner*, dejando que *Maple* elija el *ranking* más eficiente en términos computacionales. En este caso solo obtenemos un sistema no trivial:

$$\begin{aligned} \dot{x}_1^3 &= y - x_2 \\ \dot{x}_2 &= -3yx_1^2 - ky - 3y + 3x_2 \\ \dot{y} &= -ky \\ u &= -3yx_1^2 - ky - 3y + 3x_2 \quad for \quad x_1 \neq 0 \end{aligned} \tag{12}$$

4. Comprobamos que el bucle cerrado realmente cumple con las especificaciones. Recordamos que la comprobación solo es posible si el sistema es caracterizable, que por lo general no lo es. Por tanto, primero tomamos el modelo de proceso (11) junto con la acción de control u obtenida en (12) y lo descomponemos obteniendo en este caso particular el mismo sistema (12).

5. Aplicamos la prueba de pertenencia y como era de esperar, el resultado es afirmativo. En otras palabras cuando el control es aplicado al sistema original, siempre y cuando $x_1 \neq 0$, las trayectorias en bucle cerrado son compatibles con la restricción.

Finalmente, queda por resaltar las ideas de interés en el artículo mencionado [12], para dilucidar el sistema físico a controlar: el circuito genético (o producción de una proteína).

En [12] se demuestra que la célula (bacteria *E.coli* en este caso) hace uso de sus recursos como ARN polimerasa y ribososmas, para la producción de proteínas. Aunque la ARN polimerasa y los ribosomas son complejos abundantes en la célula [1], también son finitos. El rol que juegan ambos en la producción de proteínas tanto para la supervivencia de la misma célula, como una nueva proteína de interés, es de vital importancia. La expresión genética[1] describe el proceso de un gen (secuencia de ADN) para producir una proteína. La expresión genética tiene fundamentalmente dos fases: i) transcripción y ii) traducción. Transcripción es la producción de una molécula de ARN mensajero a partir de una molécula de ADN contenida en un gen. La traducción es la síntesis de una proteína, a partir de una molécula de ARN mensajero. La ARN polimerasa se pega al gen en la transcripción para generar ARN mensajero, mientras que los ribosomas son las que se unen al ARN mensajero para empezar la traducción de una proteína. Como los genes compiten por estos recursos que son limitados, aparecen interacciones no reguladas ni intencionadas que pueden

[1]Proceso por el que microorganismo procariotas y células eucariotas transforman la información codificada por los ácidos nucleicos en proteinas.

afectar dramáticamente al comportamiento del circuito genético.

Aquí también se señala que mediante un método combinado de modelado y estudio experimental, han logrado caracterizar el grado en el que la competición por los recursos (ARN polimerasa y ribosomas) afecta al comportamiento del circuito genético. Este método, consiste en modelar a través de 'coeficientes de demanda', que pueden ser ajustados por los parámetros del circuito, en tanto estos coeficientes dictaminen la intensidad de las interacciones no reguladas, permitiendo una guía efectiva del diseño de un circuito hacia el comportamiento deseado por el diseñador.

Entrando en detalle, predecir el comportamiento de circuitos genéticos en células vivas es un desafío recurrente en la biología sintética. Los circuitos genéticos son usualmente vistos como interacciones de nodos. Cada nodo esta compuesto por un núcleo de expresión genética, donde tienen lugar la transcripción y traducción de un gen. En este trabajo vemos cada nodo como un sistema de entrada / salida donde una especie bioquímica (o proteína) de entrada regula la producción de otra de salida. En la realidad, el comportamiento de un nodo a menudo depende de su entorno, incluyendo otros nodos en el mismo circuito y el medio de la célula. Este hecho limita significativamente la capacidad actual para diseñar circuitos genéticos que se comporten de la forma deseada.

La expresión genética emplea un fondo común de recursos. En particular la disponibilidad de ARN polimerasa y ribosomas ha sido identificada como el mayor cuello de botella para la expresión genética en una bacteria. Cuando un nodo es activado este merma el fondo común de recursos reduciendo, su disponibilidad para el resto de nodos en el circuito y esto puede afectar potencialmente al comportamiento conjunto del circuito.

Por lo que en [12] se busca como determinar la competición de los genes que forman el circuito genético, por los recursos como ARN polimerasa y ribosomas. Para ello los autores desarrollan un modelo matemático que explícitamente incluye la competición mediante funciones de *Hill* [2]. Estas funciones describen la interacciones regulatorias que se derivan de reacciones químicas, considerando la ley de conservación, bajo la asunción de que la concentración de ARNp y ribosomas libres es constante, debido a que su disponibilidad es limitada, obtienen el siguiente modelo de *Hill* modificado:

$$\begin{aligned} \dot{x}_1 &= G_1(x_1, x_2, I_1) - \gamma_1 x_1 \\ \dot{x}_2 &= G_2(x_1, x_2, I_2) - \gamma_2 x_2 \end{aligned} \tag{13}$$

donde x_1 y x_2 son las proteínas de interés, I_1 e I_2 son inductores y γ_1, γ_2 son los factores de degradación de la proteína. El modelo de ecuaciones diferenciales ordinarias ODE (13), nos ha servido de base para el modelado desarrollado en el presente trabajo.

De acuerdo con el modelo (13), el ratio de producción efectiva $G_i(x, I_i)$ está conjuntamente afectado de forma proporcional por la regulación de la expresión genética y por la modulación alostérica[2], y está además inversamente afectado por la competición de recursos. En particular puesto que los recursos son compartidos entre todos los nodos, $G_i(x, I_i)$ tiene un común denominador para cada ratio de producción efectiva.

Ahora que los trabajos previos han ubicado en cierta medida el área de estudio de este proyecto, en los siguientes apartados definimos con más detalle tanto el tema como la terminología ya empleada, del algoritmo de *Rosenfeld-Groebner*, de su empleo para diseño de controles y de los procesos biológicos junto al modelo matemático del circuito genético.

[2]Consiste en el cambio estructural de una proteína. Uno de sus receptores se une a un efector el cual producirá un cambio en la proteína. Los efectores pueden ser tanto positivos, favoreciendo la producción de proteína, como negativos.

2.2. Algoritmo de Rosenfeld-Groebner

Brevemente explicado, el algoritmo de *Rosenfeld-Groebner* es un algoritmo del álgebra computacional que nos permite reescribir un sistema de ecuaciones en otro sistema equivalente, con el mismo conjunto de soluciones y conservando el significado original de las variables.
El algoritmo de *Rosenfeld-Groebner* solamente es aplicable sobre conjuntos pertenecientes a un anillo diferencial polinomial[3]. En otras palabras, el modelo matemático, que describa el comportamiento del sistema a estudiar, debe estar constituido por un conjunto de ecuaciones polinomiales, con o sin derivadas y con coeficientes pertenecientes al cuerpo correspondiente del anillo.

Términos como anillo o cuerpo se desarrollan tanto matemática como intuitivamente en los siguientes subapartados.

2.2.1. Conjunto de polinomios

Como ya sabemos el algoritmo de *Rosenfeld-Groebner* solamente es aplicable sobre ecuaciones polinomiales. Las ecuaciones polinomiales son aquellas constituidas únicamente por polinomios y un polinomio se define como un conjunto finito de variables y constantes, también llamadas coeficientes, con las operaciones aritméticas suma, resta y multiplicación así como exponentes enteros positivos.
El grado de un polinomio hace referencia al mayor exponente de éste, por lo que matemáticamente un polinomio de una variable, de grado n viene descrito por la expresión:

$$P(x) = a_n x^n + a_{n-1} x^{n-1} + \dots a_0 \tag{14}$$

donde x es la variable y $\{a_n, a_{n-1}, \dots, a_0\}$ cada uno de los coeficientes del polinomio.

Durante el proyecto al conjunto de polinomios sobre el que actuamos el algoritmo lo denotamos como:

$$P = [p_1, p_2, ..., p_m] \tag{15}$$

y adelantamos que estará formado por las ecuaciones del sistema y el modelo de referencia, cuando nuestra intención sea buscar una señal de control o por las ecuaciones del sistema y la acción de control candidata cuando realicemos el test de pertenencia.

A modo ilustrativo, vamos a definir cual sería el conjunto de polinomios en un sistema de masa con resorte, junto a una dinámica objetivo cualquiera, por ejemplo: $\dot{y}(t) = -ky(t)$. El conjunto de polinomios resultante es:

$$\begin{aligned} p_1 &= \frac{1}{M}\left[u(t) - c\dot{x}(t) - kx(t) - ka^2x^3(t)\right] - \ddot{x}(t) \\ p_2 &= y(t) - x(t) \\ p_{ref} &= \dot{y} + ky(t) \end{aligned} \tag{16}$$

que bajo la notación empleada se expresaría como: $P = [p_1, p_2, p_{ref}]$.

Habiendo descrito tanto matemática como intuitivamente el concepto de polinomio, en el siguiente apartado nos disponemos a desarrollar el concepto de anillo polinomial.

[3]Es un anillo polinomial con un conjunto finito de operadores derivada δ, conmutables entre sí y lineales.

2.2.2. Anillo diferencial polinomial

El algoritmo de *Rosenfeld-Groebner* se construye entorno a las bases de *Groebner*, que en esencia son un sistema generador de un subconjunto llamado ideal denotado como I, perteneciente a un anillo polinomial. Por ello, y concretando todavía más, en este proyecto nos centramos en los anillos diferenciales polinomiales, sobre el cuerpo de los racionales, con polinomios reales y con el operador derivada total temporal:

$$A = \mathbb{Q}[x_1, ..., x_n, \dot{x}_1, ..., \dot{x}_n, ...] \tag{17}$$

Dejando para más adelante la definición y desarrollo de conceptos como ideal o base de *Groebner*, planteamos de forma cualitativa el concepto de anillo diferencial.

Podemos definir un anillo como un conjunto que trabaja con dos leyes de composición interna, usualmente el producto y la suma, que es asociativo para ambas, el producto es distributivo respecto a la suma y tiene elemento neutro tanto para la suma como para la multiplicación pero, aunque tiene inverso con la suma no lo tiene con la multiplicación. La principal diferencia con un cuerpo es esta última, un cuerpo comparte todas las características de un anillo y además tiene elemento simétrico para el producto, es decir inverso.

Por ejemplo, y de forma práctica en el conjunto de los polinomios al cumplir las propiedades mencionadas antes y especialmente la del inverso, ya que para un polinomio cualquiera no tiene porque existir otro polinomio que al multiplicarlo de como resultado el elemento neutro $e = 1$, tiene estructura de anillo.

Los apellidos polinomial y diferencial vienen por que el conjunto sobre el que se define es justamente los polinomios y por que puede contener ecuaciones diferenciales, es decir alguna derivada temporal sobre alguna variable.

Cuando hablamos de polinomios reales sobre el cuerpo de los número racionales, representados matemáticamente por el símbolo $\mathbb{Q}$, no referimos a polinomios de variable real, es decir que la variable solo toma valores reales, con coeficientes perteneciente a los números racionales.

Además, como ya sabemos, el algoritmo de *Rosenfeld-Groebner* comparte características con algoritmos de división, por lo que es necesario definir y se hace sobre el propio anillo, un orden de los monomios[4] que denominamos *ranking*. De esta forma la división entre polinomios queda definida.

Para definir el orden de monomios, que representamos mediante los símbolos $(\prec)$, $(\succ)$, en el anillo, es decir su *ranking*, nos ayudamos de la siguiente notación:

$$\text{Def: } f = \textstyle\sum_\alpha a_\alpha x^\alpha \text{ polinomio distinto de cero} \in k[x]$$

I. Multigrado de f: $multideg(f) = max(\alpha; a_\alpha \neq 0)$

II. Coeficiente principal de f: $LC(f) = a_{multideg(f)} \in k$

III. Monomio principal de f: $LM(f) = x^{multideg(f)}$

IV. Término principal de f: $LT(f) = LC(f) \cdot LM(f)$

[4]Expresión formada por una variable y un parámetro constante, con las operaciones aritméticas producto así como exponente de enteros positivos. Se denomina polinomio a la suma de monomios.

En la notación anterior no aparecen derivadas temporales ya que el criterio de ordenación para éstas siempre será el del orden del monomio:

$$\text{Si } x_1 \prec x_2 \text{ entonces } x_1 \prec \dot{x}_1 \text{ y } \dot{x}_1 \prec \dot{x}_2$$

y aunque la notación está construida sobre un polinomio de una sola variable, es extrapolable a polinomios de distintas variables.

Si bien existen varios tipos de *ranking*, en el proyecto solo nos centramos en describir los tipos de *ranking* más relevantes para el algoritmo de *Rosenfeld-Groebner*:

Orden por grado: este *ranking* suele emplearse para polinomios de una variable y ordena los monomios de mayor a menor grado:

$$\cdots > x^{m+1} > x^m > \cdots > x > 1$$

Orden lexicográfico: por definición el multigrado $\alpha >_{lex} \beta$ siendo $\alpha = (\alpha_1, \ldots, \alpha_n)$ y $\beta = (\beta_1, \ldots, \beta_n)$, cuando el término más a la izquierda de la relación $\alpha - \beta$ es mayor que 0.

$$\text{Ej: } \alpha = (1, 2, 0) >_{lex} \beta = (0, 3, 4) \rightarrow \alpha - \beta = (1, -1, -4)$$

En la práctica es usual trabajar con polinomios de más de una variable, que podemos ordenar 'alfabéticamente' usando un orden lexicográfico.

$$\text{Ej: } (1, 0, 0) >_{lex} (0, 1, 0) >_{lex} (0, 0, 1)$$
$$x >_{lex} y >_{lex} z$$

Sin embargo, a la hora de obtener bases de *Groebner* este tipo de ranking genera problemas de coste computacional, ya que prioriza ciertos monomios independientemente del grado del resto, por lo que es conveniente evitar su uso en sistemas complejos.

Orden lexicográfico graduado: emplear un orden graduado implica ordenar comparando el grado total, de mayor a menor. Esto puede generar ambigüedades que llamamos empate (cuando los grados son iguales), que solucionamos aplicando el orden lexicográfico.

Que de matemáticamente corresponde a decir que $\alpha >_{glex} \beta$ si:

$$|\alpha| = \sum_i \alpha_i > |\beta| = \sum_i \beta_i$$

$$or$$

$$|\alpha| - |\beta| = 0 \quad \wedge \quad \alpha >_{lex} \beta$$

Ej:

$$\alpha = (1, 2, 4) >_{glex} \beta = (1, 1, 5)$$
$$|\alpha| - |\beta| = 7 - 7 = 0 \quad \wedge \quad \alpha - \beta = (0, 1, -1)$$

Es precisamente una versión de este *ranking*, llamada **orden lexicográfico graduado inverso**, el que mejores resultado de costo computacional ofrece a la hora de hallar las bases de *Groebner*. La diferencia se encuentra en que mientras se mantiene el ordenado de grado total, los empates se resuelven con el orden lexicográfico inverso.

Ej:

$$Grlex(>_{glex}) : x^2z^2 > xy^2z > x^3 > z^2$$
$$Grevlex(>_{grlex}) : xy^2z > x^2z^2 > x^3 > z^2$$

Ahora que estamos familiarizados con el término anillo polinomial, el siguiente paso lógico en el desarrollo del algoritmo, es mostrar las propiedades del subconjunto del anillo que es generable a partir de la base de *Groebner* y que al inicio del subapartado hemos adelantado su nombre, el ideal matemático I.

2.2.3. Construcción del ideal diferencial y descomposición del radical

Si bien en el apéndice, concretamente en conceptos de álgebra abstracta, se da una definición matemática formal tanto de ideal como de radical del ideal, recordamos que intuitivamente un ideal I asociado a un sistema de ecuaciones polinomial es un conjunto infinito de polinomios tales que se anulan con las raíces[5] del sistema original. El término diferencial se refiere al hecho de que el operador derivada debe ser considerado, en tanto que aparecen ecuaciones diferenciales. Por último el término radical, representado matemáticamente como $\sqrt{I}$ y que al referirse a un conjunto mayor, indica que ningún polinomio, que se anule con las raíces del sistema de ecuaciones original, queda fuera.

Sabemos por el teorema de la base de *Hilbert* [5] que cada ideal $I \subseteq \mathbb{Q}[x_1, ..., x_n]$ es finitamente generado, es decir posee un número finito de generadores.

Entendemos como base de un ideal I al conjunto de polinomios linealmente independientes entre sí, capaces de generar cualquier polinomio del ideal mediante combinaciones lineales, como sumas y productos. Como ya hemos dicho, el hecho de que el número de generadores de un ideal sea finito garantiza bases finitas. Entre las posibles bases, destacamos un tipo llamadas bases de *Groebner*, por sus propiedades. La principal, y que aprovecha este algoritmo, es que garantiza un resto $r = 0$, al aplicar un algoritmo de división, para cualquier polinomio perteneciente al ideal.

Antes de continuar recordamos de forma aclaratoria, en que consiste un algoritmo de división polinomial e ilustraremos un ejemplo en pseudocódigo:

Centrándonos en polinomios de una variable, un algoritmo de división indica como un polinomio $f \in k[x]$ puede ser escrito de la forma $f = q \cdot g + r$ donde $q, r \in k[x]$ y $r = 0$ ó $grado(r) < grado(g)$. Es decir un polinomio se puede expresar como la suma del producto entre el cociente q y el divisor g más el resto r de la división.

Ejemplo: A través del siguiente algoritmo de división en pseudocódigo determina si el polinomio $f(x) = x^2 - 3x + 2$ pertenece al ideal $I = \langle x - 2 \rangle \subset \mathbb{R}[x]$. El *ranking* empleado es el orden por grado.

```
Input g,f
output q ,r
q =0 , r = f WHILE r != 0 AND LT(g) divides LT(r) DO
    q = q + LT(r)/LT(g)
    r = r - (LT(r)/LT(g))·g
```

[5]En matemáticas la palabra raíces hace referencia al conjunto de soluciones de una ecuación algebraica.

Entradas:

$$g = x - 2$$
$$r = x^2 - 3x + 2$$
$$q = 0$$

Primera iteración:

$$q = 0 + \frac{x^2}{x} = x$$
$$r = x^2 - 3x + 2 - \frac{x^2}{x}(x-2) = x^2 - 3x + 2 - x^2 + 2x = 2 - x$$

Como $r \neq 0$ realizamos una segunda iteración:

$$q = x + \frac{-x}{x} = x - 1$$
$$r = 2 - x - \frac{-x}{x}(x-2) = 2 - x + x - 2 = 0$$

Como el resto de la división es $r = 0$ y el polinomio $q \in \mathbb{R}[x]$, pertenece al anillo polinomial, concluimos que el polinomio pertenece al ideal I. Comprobación:

$$f = gq + r = (x-2)(x-1) = x^2 - x - 2x + 2 = x^2 - 3x + 2$$

Tras el inciso y siguiendo el criterio de *Buchberger* [5], que afirma:

$$G = \{g_1, ..., g_t\} \text{ es una base de } \textit{Groebner} \text{ de } \langle I \rangle \ \forall g_{i,j} \in G \ ; \ i \neq j$$
$$\text{si el residuo en la división de S-poli}(g_i, g_j) \text{ entre } G \text{ es } 0$$

podemos construir una base de *Groebner*, a través de un determinado algoritmo de división que además, en el caso que el resto de la división fuese distinto de cero, al pertenecer al ideal, el polinomio resultante puede añadirse a la base G y evaluar de nuevo el criterio de *Bucherberg* hasta que el resto sea nulo. Sabemos que en algún momento convergerá ya que como hemos visto antes los ideales polinomiales tienen un número finito de generadores.

Definición de S-polinomio: $S(f,g) = \frac{x^\gamma}{LT(f)} \cdot f - \frac{x^\gamma}{LT(g)} \cdot g$, donde $x^\gamma = m.c.m(LM(f), LM(g))$

Ej:

$$\{f = x^3y^2 + xy^3,\ g = xyz - z^3\}$$
$$LM(f) = x^3y^2,\ LM(g) = xyz,\ x^\gamma = x^3y^2z$$

$$S(f,g) = \frac{x^3y^2z}{x^3y^2}\left(x^3y^2 + xy^3\right) - \frac{x^3y^2z}{xyz}\left(xyz - z^3\right) = z\left(x^3y^2 + xy^3\right) - x^2y\left(xyz - z^3\right) = zxy^3 + x^2yz^3$$

Como vemos en el ejemplo anterior el S-polinomio depende del tipo de *ranking* escogido, lexicográfico en este caso, y por tanto el tipo de *ranking* también influirá en los polinomios que formen el ideal de la base de *Groebner*. En general las bases de Groebner son un subconjunto

del ideal del anillo polinomial $\langle G\rangle \subseteq \langle I\rangle$ por lo que, según el *ranking*, el algoritmo de *Rosenfeld-Groebner* puede generar bases de *Groebner* distintas.

A priori no conocemos ningún método que nos diga que tipo de *ranking* es el más adecuado para obtener la base de *Groebner* idónea (más simple o en la forma adecuada para nuestra búsqueda), por lo que normalmente trabajamos con *rankings* que llamamos de eliminación o de bloque, que usan el orden lexicográfico y que dentro de lo posible triangularizan el sistema, aunque en ocasiones aumenten la complejidad de las expresiones frente otros *rankings*.

Finalmente podemos descomponer el ideal en su radical $\sqrt{I}$, que está formado por cualquier potencia entera y positiva de los polinomios que generan el ideal: $\{f : f^m \in I; m \geq 1\}$. Por definición un ideal es un subconjunto de su radical $I \subset \sqrt{I}$ y a continuación vamos a demostrar a que nos referimos con un ejemplo práctico:

Ej: si consideramos el ideal $J = \langle x^2, y^3\rangle \subset K[x, y]$. Por definición:

$$x, y \notin J$$
$$x, y \in \sqrt{J}$$

Mientras que $x^2y^2 \in J$ ya que $x^2 \in J$ e $y^2 \in K[x, y]$, se da el caso que $xy \notin J$ ya que ni x, ni y pertenecen al ideal. En cambio, tanto xy como $x^2y^2 = (xy)^2$ sí pertenecen al radical puesto que x e y también lo hacen.

El radical del ideal $\sqrt{I}$ puede relacionarse con la variedad algebraica formada por el conjunto de soluciones del sistema. Justificado por el teorema fuerte de *Nullstellensatz* [5]:

$$I(V(I)) = \sqrt{I} \tag{18}$$

siempre y cuando el cuerpo del anillo diferencial sea algebraicamente cerrado[6]. Cualitativamente nos indica una relación entre polinomios con el mismo conjunto de soluciones, es decir sistemas equivalentes y además expresados con las mismas variables.

Remarcar que aunque el cuerpo de los números racionales $\mathbb{Q}$ no es un cuerpo algebraicamente cerrado, su clausura algebraica, los números algebraicos[7] si lo es y éste justamente es el conjunto con el que trabaja el algoritmo de *Rosenfeld-Groebner*.

Para asentar los pasos del algoritmo e ilustrar como usándolo podemos diseñar un control invariante, en el siguiente apartado vamos a aplicarlo sobre un ejemplo matemático al que imponemos una dinámica objetivo.

2.2.4. Ejemplo práctico

Con ayuda de las herramientas que utilizamos a lo largo del proyecto *Maple* y *SIMULINK* vamos a diseñar un control sobre un sistema estilo *Lotka-Volterra* de dos poblaciones, presa y predador, de forma que la población de predadores evolucione siguiendo la dinámica un oscilador amortiguado.

Matemáticamente un modelo estilo *Lotka-Volterra* expresa la dinámica entre poblaciones mediante ecuaciones polinomiales que además son no lineales, debido al producto entre variables.

[6] Un cuerpo es algebraicamente cerrado si no tiene extensiones algebraicas propias.

[7] Cualquier número que es solución de una ecuación algebraica de la forma: $a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0 = 0$

Como ya sabemos, por sus características, el sistema de ecuaciones es tratable con el algoritmo de *Rosenfeld-Groebner*.

Modelo de *Lotka-Volterra*:

$$\dot{x}_1(t) = x_1(t)\,(a - b \cdot x_2(t)) + u(t) \tag{19}$$

$$\dot{x}_2(t) = -x_2(t)\,(c - d \cdot x_1(t)) \tag{20}$$

Donde:

- La variable x_1 hace referencia a la población de presas y x_2 a la de predadores.
- a,b,c y d son los parámetros de interacción entre especies.
- u es la entrada del sistema, permite la introducción o extracción de individuos en la población de presas.

La dinámica de un oscilador amortiguado, matemáticamente se puede expresar a través de una ecuación diferencial, que además es polinomial. Como hemos mencionado previamente buscamos que la población de predadores oscile de forma controlada hasta un valor concreto, que en forma de ecuación luce:

$$\ddot{x}_2(t) = -k_1 \cdot \dot{x}_2(t) - k_2 \cdot x_2(t) + k_3 \tag{21}$$

ajustable con los siguientes parámetros, donde $[T]$ hace referencia a una unidad de tiempo genérica:

- k_1 constante de amortiguamiento $[T]^{-2}$.
- k_2 constante elástica $[T]^{-1}$.
- $\frac{k_3}{k_2}$ valor final de la población de predadores.

En primer lugar, expresamos el conjunto de polinomios diferenciales, formado por las ecuaciones de *Lotka-Volterra* (19),(20) y el modelo de referencia (21), en lenguaje de *Maple*.

Ecuaciones de Lotka-Volterra
$p1 := diff(x_1(t), t) - x_1(t) \cdot (a - b \cdot x_2(t)) - u(t) :$
$p2 := diff(x_2(t), t) + x_2(t) \cdot (c - d \cdot x_1(t)) :$

Modelo de referencia
$referencia := diff(diff(x_2(t), t), t) + k_1 \cdot diff(x_2(t), t) + k_2 \cdot x_2(t) - k_3 :$

Conjunto de polinomios
$P := [p1, p2, referencia] :$

Llamamos a la librería de álgebra diferencial, esto permite acceso a *Maple* a las herramientas matemáticas que participan en el algoritmo de *Rosenfeld-Groebner*.

Librería de algebra diferencial
$with(DifferentialAlgebra)$

Al generar el anillo diferencial, tenemos que indicar el tipo de *ranking* (*blocks*), el tipo de operador derivada, total o parcial, y respecto a que variable (*derivations*) e indicar, si hay parámetros libres, su pertenencia al cuerpo del anillo (*arbitrary*).

Notaciones de ranking en *Maple*:

- $blocks = [x_1, x_2, x_3]$ es la notación de un *ranking* de eliminación: $x_1 = x_1(x_2, x_3)$, $x_2 = x_2(x_3)$
- $blocks = [[x_1, x_2, x_3]]$ indica a *Maple* que escoja el *ranking* que genere la base con polinomios de menor grado.
- $blocks = [x_1, [x_2, x_3]]$ es la notación que indica la combinación de los *rankings* anteriores donde: x_2 y x_3 siguen bajo un *ranking* de eliminacion en el que $x_2 = x_2(x_3)$ pero la expresión de x_1 no tiene porque triangularizar el sistema.

Anillo diferencial
$R := DifferentialRing(blocks = [u, [x_1, x_2]], derivations = [t], arbitrary = [k_1, k_2, k_3])$

Algoritmo de Rosenfeld-Groebner
$G := RosenfeldGroebner(P, R)$

Al obtener la variable G que es el ideal $\langle G \rangle$ que contiene la base de *Groebner*, termina la aplicación del algoritmo.

Por el algoritmo sabemos que el conjunto de polinomios que conforman G es equivalente al conjunto P, en el sentido que comparten conjunto de soluciones y el significado de las variables. El siguiente paso es escoger uno de los elementos de la base G, un polinomio, el que convenga por simplicidad o por dependencia entre variables, como candidato a acción de control del sistema. En nuestro caso, como hemos empleado un *ranking* de eliminación para reescribir el sistema de forma que la entrada quede en función de las dos poblaciones, escogemos:

$$u(t) = -\frac{1}{x_2(t)d}\left(x_1(t)^2x_2(t)d^2 - x_1(t)x_2(t)^2bd + x_1(t)x_2(t)k_1d + x_1(t)x_2(t)ad - 2x_1(t)x_2(t)cd - x_2(t)k_1c + k_2x_2(t) + x_2(t)c^2 - k_3\right) \quad (22)$$

El siguiente paso es comprobar que el controlador descrito por la acción de control (22) y las ecuaciones del sistema (19), (20), es capaz de reproducir el modelo de referencia. Para ello aplicamos el algoritmo de *Rosenfeld-Groebner* sobre el conjunto de ecuaciones del controlador, obteniendo una nueva base de *Groebner* a la que llamamos GC. Si la ecuación dinámica del modelo de referencia (21) es generable por esta nueva base, quiere decir que el controlador es válido.

$PC := [p1, p2, control]$:
$GC := RosenfeldGroebner(PC, R)$:
$BelongsTo(referencia, GC)$
$> true$

En este caso la referencia pertenece a la nueva base *Groebner* y por tanto la acción de control candidata, es válida para el diseño del control.

Simulación en MATLAB-SIMULINK

Para simular como actúa la acción del control sobre el sistema de *Lotka-Volterra* usamos la herramienta *SIMULINK* que nos permite programar la dinámica del sistema mediante bloques y realimentaciones:

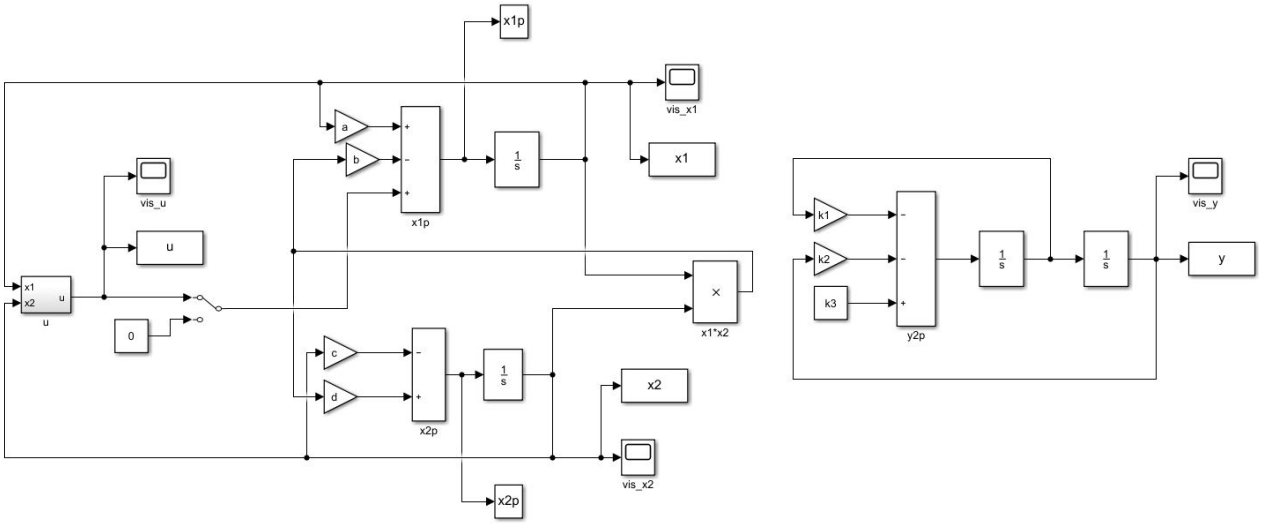


Figura 2: Esquema de *SIMULINK* de LV con acción de control + esquema de modelo de referencia.

El esquema de *SIMULINK* de la figura (2) está distribuido en dos regiones diferenciadas. En la zona izquierda se encuentra el sistema de *Lotka-Volterra* implementado junto a la acción de control u, mientras que en la derecha está implementado, de forma paralela y para comparar posteriormente, el modelo de referencia del oscilador amortiguado donde hemos renombrado la variable x_2 por y para ejecutarlos de forma independiente y así posteriormente poder comprobar si el comportamiento coincide.

En primer lugar, simulamos el comportamiento del sistema de *Lotka-Volterra* sin acción de control ($u(t) = 0$) y con los siguientes parámetros y condiciones iniciales:

$$\begin{aligned}[a, b, c, d] &= [0{,}1; 0{,}02; 0{,}3; 0{,}01] \text{ con unidades de frecuencia } [T]^{-1} \\ x_1(0) &= 40 \\ x_2(0) &= 9\end{aligned}$$

para posteriormente apreciar el efecto de la acción de control sobre el sistema.

Como vemos en la figura (3) la población de una especie influye en la población de la otra creando una variación cíclica. Cuando la población de presa aumenta, al haber más alimento para los predadores crece la población de estos últimos, reduciendo la población de presas debido al aumento de caza y por tanto reduciendo el alimento de los predadores y su población de nuevo.

El tiempo está expresado de forma escalable a las unidades de los parámetros, así vemos el comportamiento general, independientemente del tipo de presas y predadores.

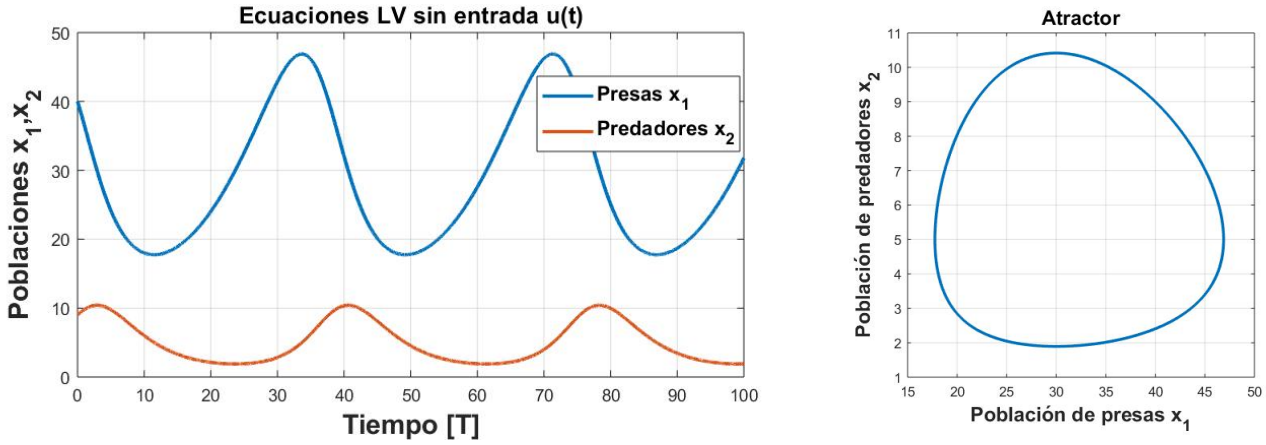


Figura 3: Simulación de LV sin acción de control.

Con la acción de control obtenida podemos, por ejemplo, estabilizar la población de predadores en un valor concreto. Para ello basta con ajustar los valores de los parámetros del modelo de referencia (21), según el comportamiento deseado. Tomando los valores: $k_1 = k_2 = 1$ definimos el tipo de oscilación y de amortiguación, mientras que el último parámetro nos sirve para indicar la cantidad de población de predadores en la que se estabilizará el sistema.

Cuando el sistema es estable $\ddot{x} = \dot{x} = 0$ la población de predadores se estabiliza en:

$$0 = -k_1 \cdot 0 - k_2 x_2 + k_3 \rightarrow \quad x_2 = \frac{k_3}{k_2}$$

por lo que si buscamos, por ejemplo, que la población de predadores se estabilice en 25 unidades, basta con ajustar el parámetro k_3, a partir de la relación:

$$25 = \frac{k_3}{k_2} \rightarrow \quad k_3 = 25 \cdot k_2 = 25 \cdot 1 = 25$$

Veamos, en la siguiente simulación, como actúa el control a partir del ajuste de parámetros en la señal de control u mencionado:

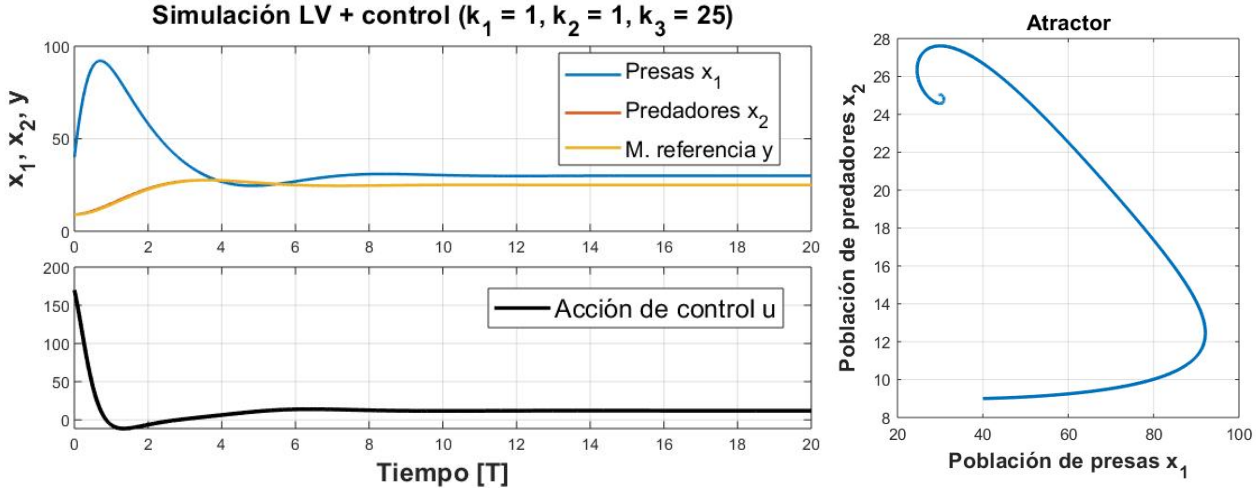


Figura 4: Simulación de LV con acción de control.

A primera vista, la figura (4) muestra un comportamiento en el que la población de predadores además de seguir las dinámica de referencia, se estabiliza en un valor concreto. Esto es un

buen indicio de que el controlador funciona correctamente. Yendo más allá si bien es complicado comprobar gráficamente la influencia de los parámetros k_1 y k_2 correspondientes al amortiguamiento y a la constante elástica respectivamente, es relativamente sencillo en el caso del término $\frac{k_3}{k_2}$, bastando con comprobar si la población de predadores se estabiliza en el valor: $\frac{k_3}{k_2} = \frac{25}{1} = 1$ y efectivamente, así es.

Llegados a este punto, en el que conocemos el funcionamiento del algoritmo de *Rosenfeld-Groebner* y como aplicarlo para el diseño de controles invariantes. Nos disponemos a presentar el circuito genético, tanto cualitativa, sobretodo en los procesos biológicos involucrados, como matemáticamente a la hora de modelar su comportamiento.

2.3. Circuito genético

En biología sintética, el uso de ADN codificado en genes que tienen estructura definida, dentro de un célula, de forma que pueda desarrollar funciones lógicas y conseguir un comportamiento deseado a partir de unos valores de entrada, se denomina circuito genético.

El circuito genético tratado en este proyecto consta de dos genes: gA y gB, que a su vez, producen las proteínas A y B respectivamente. Ellos interactúan con un pozo común de ARN polimerasa y ribosomas para iniciar la expresión genética. Como se dijo anteriormente estos recursos son abundantes dentro de la célula, pero limitados. En particular, son los ribosomas[8] y la polimerasa[9] los recursos que generan el mayor cuello de botella identificado en la expresión genética dentro de una célula. Nuestro circuito genético se construirá en la bacteria *E.coli* que es una de las células procariotas más utilizadas, por lo que es necesario considerar este cuello de botella.

Paralelamente, la forma de actuar sobre el circuito son los ya comentados valores de entrada o inductores. Los inductores son especies bioquímicas cuyas moléculas se adhieren al promotor de un gen (secuencia específica de ADN), para iniciar la transcripción de un gen y consecuentemente producir una proteína.

Para clarificar conceptos, mostramos de forma esquemática el circuito genético, figura (5):

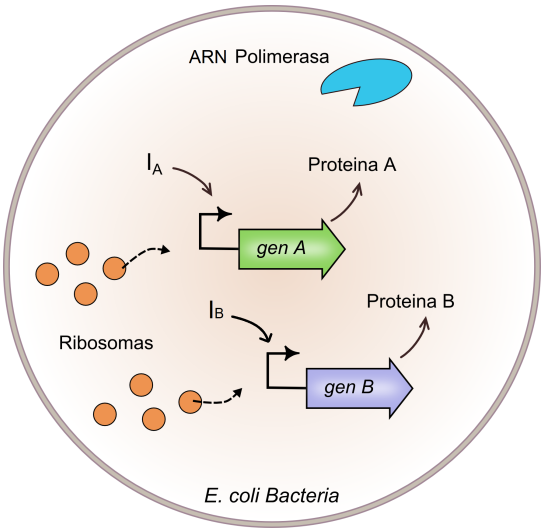


Figura 5: Circuito genético.

[8]Complejos macromoleculares de proteína.
[9]Enzima compleja capaz de replicar los ácidos nucleicos (ADN, ARN).

donde podemos observar una bacteria *E.coli* que alberga los dos genes gA y gB, los recursos de la propia célula que son los ribosomas y la ARN polimerasa, que son empleados por los genes para producir proteínas A y B y los inductores I_A e I_B que son las entradas del circuito genético.

Analizando la expresión del gen gA, que a su vez es análogo al del gen gB, podemos entender mejor su comportamiento, como interactúan entre sí los genes mediante la competición por recursos y a su vez extraer un modelo matemático del circuito que recoja todas las características de interés.

2.3.1. Proceso biológico

Una cadena de ADN se divide en unidades funcionales llamadas genes. Cada gen proporciona las instrucciones para formar una proteína, ya sea para desempeñar un trabajo en la célula o simplemente una proteína de interés que queramos producir. Estas proteínas, se conocen también como polipéptidos[10]. Aunque muchas proteínas se conforman de un solo polipéptido, algunas están hechas de varios de ellos. Los genes que especifícan polipéptidos se conocen como genes codificantes de proteínas. Aunque no todos los genes codifican proteínas, algunos proporcionan instrucciones para producir moléculas de ARN funcionales como los ARN de transferencia, que desempeñan un papel en la traducción. Pero, ¿cómo dirige exactamente el ADN la construcción de un polipéptido?

Brevemente como se dijo anteriormente, este proceso consta de dos pasos: i) la transcripción, llamado así porque implica 'transcribir' la secuencia de ADN en un 'alfabeto' de ARN similar, y ii) la traducción donde la secuencia de nucleótidos[11] del ARN mensajero se debe 'traducir' al 'lenguaje' de los aminoácidos.

Durante la expresión de un gen codificante de proteína, la información fluye:

$$\text{ADN} \rightarrow \text{ARN mensajero} \rightarrow \text{proteína}$$

este flujo de información se conoce como el **dogma central** de la biología molecular [1].

La transcripción es el primer paso de la expresión genética. El objetivo de la transcripción es producir una copia de ARN mensajero de la secuencia de ADN de un gen. En el caso de los genes codificantes de proteína, la copia de ARN mensajero, o transcrito, contiene la información necesaria para generar un polipéptido. La principal enzima que participa en la transcripción es la ARN polimerasa, que a partir de un molde de ADN genera una cadena complementaria, con la misma secuencia de información del ADN. La transcripción de un gen ocurre en tres etapas:

1) **Iniciación**: la polimerasa se une a una secuencia de ADN llamada promotor, que se encuentra al inicio del gen. Una vez unida la polimerasa separa las cadenas de ADN para proporcionar el molde necesario para la transcripción.

2) **Elongación**: con la ayuda del molde la polimerasa produce una molécula de ARN a partir de nucleótidos complementarios.

3) **Terminación**: las secuencias de ADN llamadas terminadores indican que se ha completado el transcrito de ARN. Una vez completado la copia de ARN es liberada de la polimerasa. En bacterias, las copias de ARN pueden actuar como ARN mensajeros.

Finalizada la transcripción comienza la traducción, durante ésta el ARN mensajero proporciona las instrucciones para formar un polipéptido o proteína funcional. Los ribosomas son las estructuras

[10]Un polipéptido es una cadena de aminoácidos.

[11]Molécula orgánica formada, entre otros, por las bases nitrogenadas (A,G,T,C,U).

donde se construyen los polipéptidos. Al igual que con la transcripción, la traducción se puede dividir en 3 etapas:

1) **Comienzo**: el ribosoma se ensambla alrededor del ARN mensajero. Este conjunto es conocido como conjunto de iniciación.

2) La **extensión** de la cadena: durante esta etapa la cadena de aminoácidos se extiende, agregando el aminoácido correspondiente a cada codón[12] de la cadena creciente de proteína.

3) **Finalizado**: es la etapa donde la cadena polipeptídica completa o complejo ribosómico, es liberada. Comienza cuando un codón de terminación entra al ribosoma, lo que dispara una serie de eventos que separa la cadena.

Resumiendo conceptos y de forma esquemática, el dogma central de la biología se muestra en la figura (6).

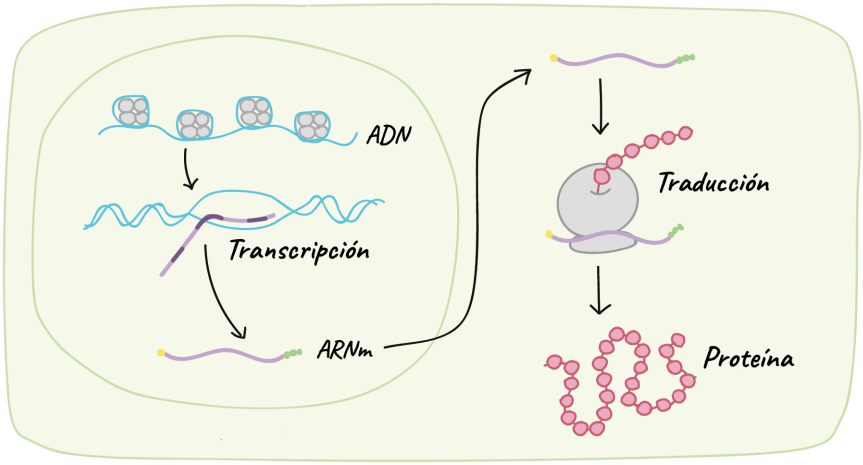


Figura 6: Expresión génica.

En el circuito genético del proyecto, descrito en la figura (5), tienen lugar durante las siguientes reacciones bioquímicas: la expresión génica del gen gA (análoga a la del gen gB):

1) Cuando la polimerasa (y) del medio entra en contacto con el gen gA puede asociarse a éste. A su vez esta reacción puede ocurrir en sentido contrario donde la polimerasa se disocia del conjunto ($C_A^{y_0}$) :

$$r_1 : \qquad gA + y \underset{K_{-A}^{y_0}}{\overset{K_A^{y_0}}{\rightleftharpoons}} gA \cdot y \triangleq C_A^{y_0}$$

$K_{-A}^{y_0}$ y $K_A^{y_0}$ son las constantes de disociación / asociación entre el gen gA y la polimerasa.

2) La polimerasa se une al promotor del gen gA, generando un molde para la transcripción y dando lugar a la etapa de elongación donde se produce la cadena de ARN mensajero (mA). Finalizada la transcripción la polimerasa se libera del gen y vuelve al medio.

$$r_2 : \qquad C_A^{y_0} \xrightarrow{K_{mA}} y + gA + mA$$

K_{mA} es la tasa de transcripción que hace referencia a la contribución de la etapa de elongación

[12]Grupo de 3 bases nitrogenadas.

y esta reacción es unidireccional.

3) La presencia de inductor I_A en el circuito permite regular la transcripción, ya que éste puede unirse al gen gA. Esta unión se denomina factor de transcripción (TF por sus siglas en inglés) o factor de unión (C_A^{TF}):

$$r_3: \qquad gA + n_A \cdot I_A \underset{K_A^-}{\overset{K_A^+}{\rightleftharpoons}} gA \cdot I_A^{n_A} \triangleq C_A^{TF}$$

siendo n_A el coeficiente de Hill[13] y K_A^+, K_A^- las tasas de asociación y disociación respectivamente.

4) El factor de transcripción (C_A^{TF}) promueve, como un activador, la unión del ADN con la polimerasa en la reacción r_4 y luego se inicia la transcripción de C_A^y en r_5:

$$r_4: \qquad C_A^{TF} + y \underset{K_{-A}^y}{\overset{K_A^y}{\rightleftharpoons}} C_A^y$$

$$r_5: \qquad C_A^y \xrightarrow{K_{mA}} y + C_A^{TF} + m_A$$

5) Finalizada la transcripción, son finalmente los ribosomas (z) los que se encargan de la traducción del ARN mensajero (mA) que genera la proteína A; con sus respectivas tasas de asociación / disociación (K_A^z / K_{-A}^z) en r_6 y a una tasa de traducción Kp_A.

$$r_6: \qquad m_A + z \underset{K_{-A}^z}{\overset{K_A^z}{\rightleftharpoons}} M_A^z$$

$$r_7: \qquad M_A^z \xrightarrow{K_{pA}} m_A + z + A$$

6) Por último tanto el complejo ribosómico (M_A^z) como el ARN mensajero y la proteína sintetizada tienen unas tasas de degradación:

$$r_8: \qquad M_A^z \xrightarrow{\omega_A} \emptyset$$

$$r_9: \qquad m_A \xrightarrow{d_{mA}} \emptyset$$

$$r_{10}: \qquad A \xrightarrow{d_A} \emptyset$$

2.3.2. Sistema de ecuaciones

Obtenemos las ecuaciones ODE del modelo, aplicando la ley cinética de acción de masas [10] y la aproximación *Quasy steady state*, abreviada como *QSSA* [10]. Esta aproximación consiste en aprovechar la diferencia entre procesos más rápidos frente a otros más lentos, considerando los más rápidos como si estuviesen en régimen permanente. De esta forma podemos reducir el número de variables que intervienen en el modelo, manteniendo la esencia de la dinámica del sistema.

[13]Estima el grado de cooperatividad en el proceso de unión, es decir cuantas moléculas de cierta especie deben actuar en el proceso

El modelo matemático del circuito genético viene expresado a través de dos ecuaciones diferenciales polinomiales:

$$\dot{A} = \frac{K_{pA} \cdot K_{mA} \cdot c_A}{d_{mA} \cdot \nu_A \cdot K_A} \cdot z \cdot y \cdot f_A - d_A \cdot A \tag{23}$$

$$\dot{B} = \frac{K_{pB} \cdot K_{mB} \cdot c_B}{d_{mB} \cdot \nu_B \cdot K_B} \cdot z \cdot y \cdot f_B - d_B \cdot B \tag{24}$$

donde:

$$\begin{aligned} f_A &= \frac{\frac{K_A}{K_A^0} + \frac{1}{K_{nA}} \cdot I_A^{nA}}{1 + \frac{1}{K_{nA}} \cdot I_A^{nA}} \\ f_B &= \frac{\frac{K_B}{K_B^0} + \frac{1}{K_{nB}} \cdot I_B^{nB}}{1 + \frac{1}{K_{nB}} \cdot I_B^{nB}} \end{aligned} \tag{25}$$

f_A y f_B corresponden a un tipo de funciones de *Hill* de forma sigmoidea, en este caso activadores, que representan la influencia de la concentración de los inductores (I_A, I_B) en la transcripción genética de los genes gA y gB.

$$z \cdot y = \frac{z_T \cdot y_T}{1 + J_A \cdot f_A + J_B \cdot f_B} \tag{26}$$

El producto $z \cdot y$ representa la cantidad de recursos disponibles en el medio. Los téminos del denominador indican la cantidad de recursos que está consumiendo cada uno de los genes en ese momento, donde:

$$\begin{aligned} J_A &= \frac{C_A}{K_A} \left(1 + \frac{K_{mA} \cdot y_T}{d_{mA} \cdot \nu_A}\right) \\ J_B &= \frac{C_B}{K_B} \left(1 + \frac{K_{mB} \cdot y_T}{d_{mB} \cdot \nu_B}\right) \end{aligned} \tag{27}$$

J_A y J_B son parámetros adimensionales que indican la demanda máxima de recursos por cada gen, gA y gB.

Los valores de los parámetros [3] que definen el circuito genético, vienen recopilados en la siguiente tabla:

				Descripción
$\mathbf{y_T}$		200 nM		Concentración total de polimerasa accesible.
$\mathbf{z_T}$		140 nM		Concentración total de ribosomas accesible.
Circuito A		**Circuito B**		
$\mathbf{Kp_A}$	100/60 hr^{-1}	$\mathbf{Kp_B}$	100/60 hr^{-1}	Constante de traducción.
$\mathbf{Km_A}$	100/60 hr^{-1}	$\mathbf{Km_B}$	100/60 hr^{-1}	Constante de transcripción.
cA	60 nM	**cB**	60 nM	Concentración total de ADN.
$\mathbf{dm_A}$	10/60 hr^{-1}	$\mathbf{dm_B}$	10/60 hr^{-1}	Tasa de degradación del ARN mensajero.
$\mathbf{d_A}$	0.4/60 hr^{-1}	$\mathbf{d_B}$	0.4/60 hr^{-1}	Tasa de degradación de la proteína.
$\mathbf{n_A}$	2	$\mathbf{n_B}$	2	Coeficiente de Hill.
$\mathbf{K_{0A}}$	1 μM	$\mathbf{K_{0B}}$	1 μM	Constante de disociación entre genes y polimerasa (no específica).
$\mathbf{K_A}$	500 nM	$\mathbf{K_B}$	500 nM	Constante de disociación entre genes y polimerasa (específica).
$\mathbf{K_{nA}}$	10^{n_A} nM	$\mathbf{K_{nB}}$	10^{n_B} nM	Constante de disociación entre genes y FTs.
$\nu_\mathbf{A}$	15 μM	$\nu_\mathbf{B}$	15 μM	Constante de disociación entre ribosomas y ARNm.

Tabla 1: Valores de los parámetros del modelo.

3. Desarrollo y resultados

3.1. Modelo genético

3.1.1. Diseño del control

El objetivo de este control es: intervenir en el circuito genético introduciendo una especie bioquímica, que actúe como inductor, de forma que variando su concentración en el sistema la producción de proteínas siga un comportamiento determinado.

3.1.1.1. Requisitos de diseño. Los requisitos de diseño que ha de cumplir el sistema de control son:

- El sistema ha de comportarse de acuerdo al modelo de referencia:
$$B(t) = \dot{A}(t) + \lambda \cdot A \tag{28}$$
- Una concentración del inductor del gen A (I_A) cualquiera a lo largo del tiempo.
- Construcción de la acción de control sobre la concentración del inductor con dependencia explícita de las dos proteínas $I_B = I_B(B, A)$.

Puesto que la concentración del inductor del gen B (I_B) es una sustancia que interviene directamente en la producción de proteína B, al variar la concentración de inductor estamos actuando directamente sobre la concentración de proteína.
Teniendo en cuenta que I_B es entrada del sistema, nos hace interpretar el modelo de referencia (28) como un derivador. Esto es debido a que la acción de control resultante modifica la concentración de la proteína B de forma que siga la derivada, más un término proporcional, de la concentración de la proteína A. Es decir, se trata de un control proporcional derivativo conocido como PD. Dentro del control PD podemos encontrar el caso particular del derivador puro (para $\lambda = 0$). Sin embargo, su dinámica de referencia es tan exigente que el sistema no puede responder correctamente, por ello estudiamos en su lugar un PD, que siempre podemos aproximar lo máximo que permita la dinámica del sistema, reduciendo el parámetro proporcional λ, a un derivador puro.

3.1.1.2. Empleo de herramientas basadas en álgebra diferencial. Para obtener la acción de control que cumpla con los requisitos de diseño nos ayudamos de *Maple*, en concreto de su librería de álgebra diferencial, para aplicar el algoritmo de *Rosenfeld-Groebner* al conjunto de polinomios formado por el modelo matemático del circuito genético (23), (24) y el modelo de referencia (28).

Definimos el *ranking* que mejor optimice los recursos de *Maple* para hallar la base polinomial. Si los polinomios de los ideales de la base obtenida no tuviesen la dependencia explicita deseada, emplearíamos un ranking de eliminación que forzase al algoritmo a hallar una base más adecuada.
En nuestro caso, el *ranking* más óptimo, computacionalmente hablando, genera una base de *Groebner* con un polinomio candidato a acción de control, que expresado de forma racional:

$$I_B(t)^2 = -50\frac{(-45150A(t)I_A(t)^2\lambda + 45150B(t)I_A(t)^2 + 301A(t)I_A(t)^2 - 4260000\lambda A(t) - 140000I_A(t)^2 + 4260000B(t) + 28400A(t) - 7000000)}{-23850A(t)I_A(t)^2\lambda + 23850B(t)I_A(t)^2 + 159A(t)I_A(t)^2 - 2257500\lambda A(t) - 70000I_A(t)^2 + 2257500B(t) + 15050A(t) - 3500000} \tag{29}$$

y puesto que el segundo requisito de diseño, es que la concentración del inductor del gen gA (I_A) pueda ser una señal cualquiera a lo largo del tiempo, que además se modifica desde el exterior

del sistema, la acción de control tiene una dependencia explícita, en términos de observables del sistema, de la forma:

$$I_B = I_B(A, B) \tag{30}$$

cumpliendo así el tercer requisito de diseño.

Por último para validar que la acción de control produce el comportamiento deseado, aplicamos el algoritmo de $Rosenfeld - Groebner$ al conjunto de polinomios formado por el modelo matemático del circuito genético (23), (24) y la acción de control candidata (29).
Si el polinomio del modelo de referencia (28) pertenece a la nueva base de *Groebner* quiere decir que es obtenible a partir de una combinación entre los polinomios que la forman, es decir, la acción de control junto al modelo matemático reproducen el modelo de referencia.

Y efectivamente, tras aplicar el test de pertenencia, la acción de control (29) es válida para nuestro control. En el apéndice, más concreto en programas del modelo genético, se encuentra el programa de *Maple* comentado.

3.1.2. Simulación

3.1.2.1. Esquema en SIMULINK. Simulamos el comportamiento del circuito genético junto con las entradas al sistema dadas por la concentración de los inductores I_A e I_B, que es la entrada del sistema de acuerdo a la acción de control (29), con la herramienta *MATLAB-SIMULINK*:

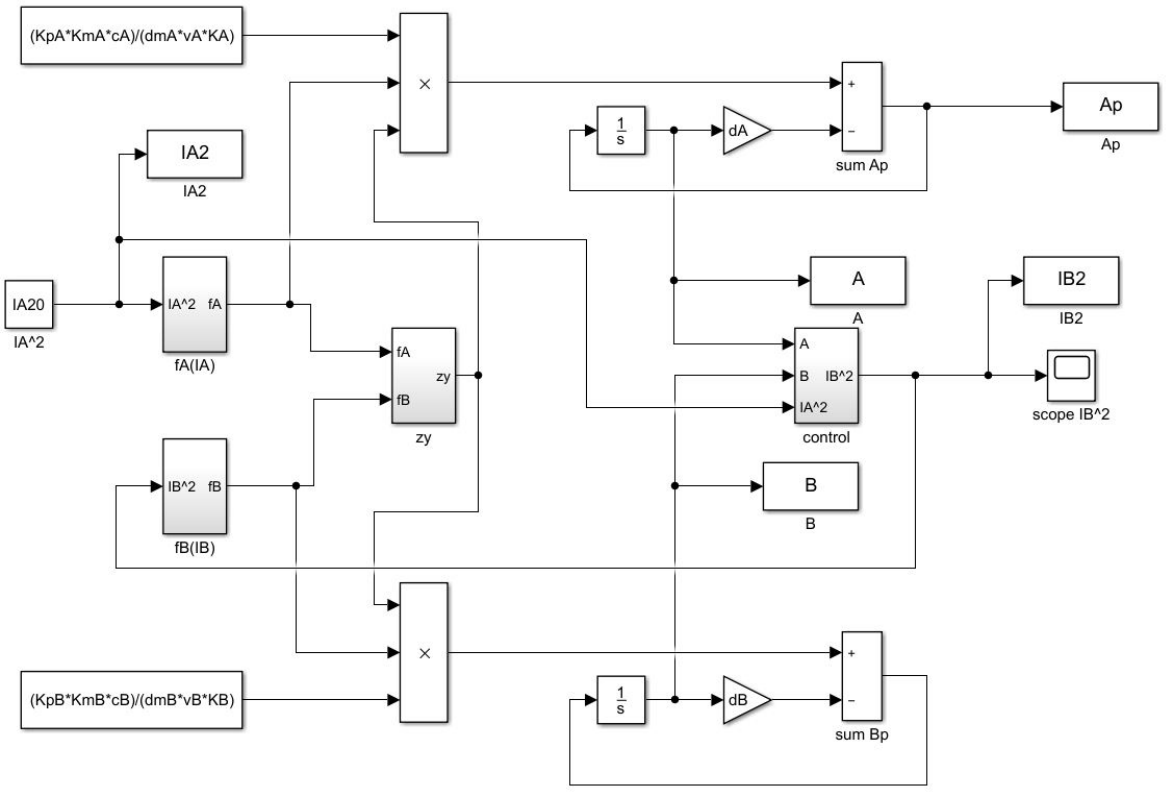


Figura 7: Esquema de *SIMULINK* del circuito genético junto a acción de control.

Un detalle a tener en cuenta es que para la simulación, por simplicidad, hemos escogido una concentración de inductor A (I_A) constante a lo largo del tiempo, esto puede apreciarse en la figura (7), donde la señal $I_A^2(t)$ proviene de un bloque del tipo constante.

3.1.2.2. Gráficas. La información visual acerca de la simulación de la acción de control sobre el modelo genético, nos permite analizar el control con más profundidad.

Si bien, la acción de control obtenida es matemáticamente válida dentro del cuerpo de los complejos $\mathbb{C}$[14], nuestras señales, al tratarse de concentraciones, solamente tienen sentido físico en el cuerpo de los reales positivos $\mathbb{R}+$.

Es por ello que, para ciertas condiciones iniciales o valores del parámetro proporcional del modelo de referencia λ, aunque matemáticamente el sistema se comporte de acuerdo al modelo de referencia, físicamente no puede hacerlo ya que las concentraciones toman valores negativos o complejos. A continuación mostramos algunos ejemplos:

Con los siguientes valores iniciales, por ejemplo, podemos conseguir concentraciones positivas, para todo instante de tiempo, según ciertos valores del parámetro λ:

$$\begin{aligned} B(0) &= 3 \quad nM \\ A(0) &= 0 \quad nM \\ I_A^2(0) &= I_A^2(t) = 1500 \quad nM^2 \end{aligned} \tag{31}$$

cuanto más cercano a cero sea el parámetro λ del modelo de referencia (28), más cercano será el comportamiento a un derivador exacto, sin embargo, al variar este parámetro, como hemos comentado antes, es posible que la respuesta física no se encuentre dentro del cuerpo de los reales positivos $\mathbb{R}+$ y por tanto no sea realizable.

Ilustremos el comportamiento mencionado con los siguientes casos prácticos. En primer lugar, para valores del parámetro λ del modelo de referencia que permitan un control en el que las variables mantengan un significado físico, podemos observar que:

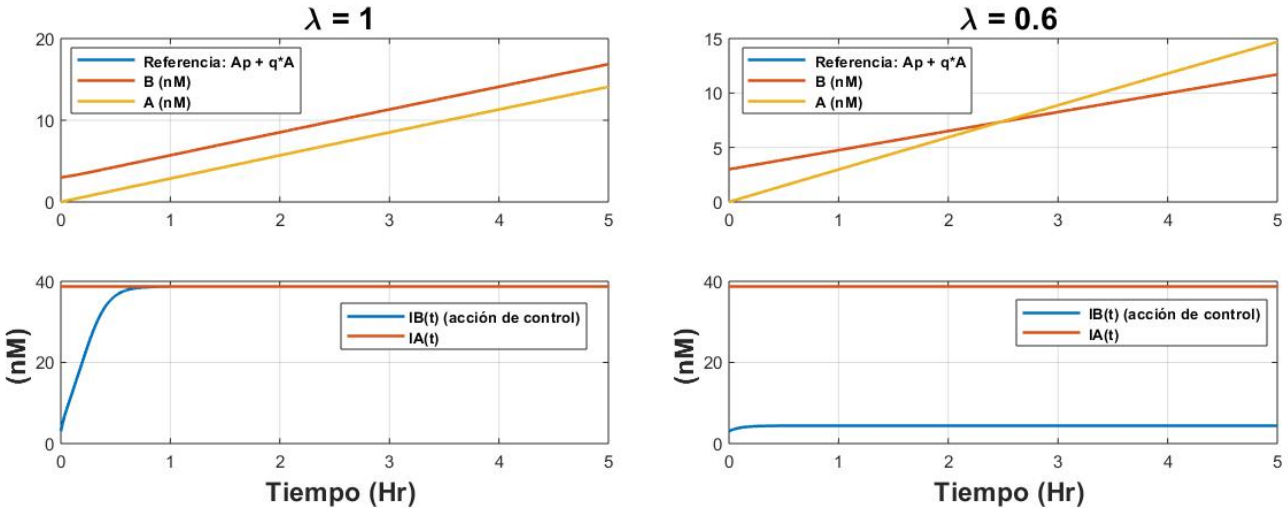


Figura 8: Ejemplos de control, físicamente realizable, con variaciones en el parámetro λ.

en el caso de la figura (8), la curva de la concentración de la proteína B sigue la referencia, está superpuesta a ésta. Vemos también que todas las concentraciones graficadas son reales y positivas y lo son para todo instante de tiempo, ya que en ambos casos, para $\lambda = 1$ y $\lambda = 0{,}6$ la concentración de inductores se estabiliza y las concentraciones de proteínas son estrictamente crecientes.

[14]Los números algebraicos (clausura algebraica del cuerpo $\mathbb{Q}$ de nuestro anillo) pertenecen al cuerpo de los complejos.

Comentar que el parámetro q que aparece en las leyendas de las gráficas en la figura (8), corresponde al parámetro λ del modelo de referencia (28), renombrado por conveniencia a la hora de simular.

En segundo lugar y como ya hemos comentado, a partir de las mismas concentraciones iniciales que en el ejemplo anterior, descritas en las ecuaciones de equivalencia (31), existen ciertos valores del parámetros λ para los que se encuentran comportamientos del sistema irrealizables, debido a que las variables pierden si significado físico. Mostramos dos casos prácticos:

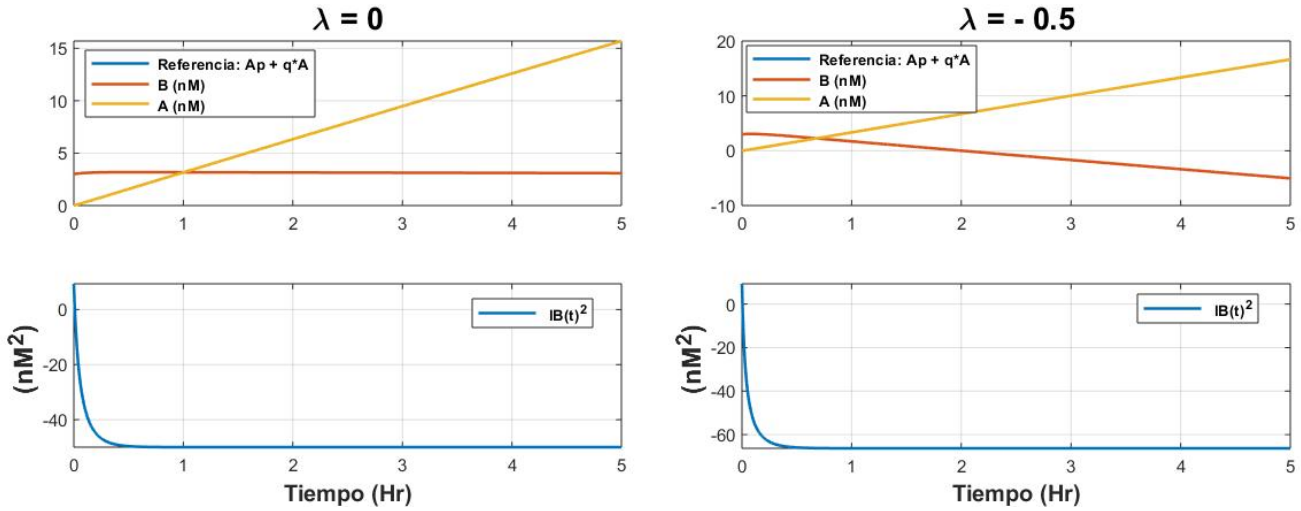


Figura 9: Ejemplos de control, físicamente imposibles.

evaluados en las gráficas de la figura (9). Si bien la concentración de proteína B, matemáticamente se comporta de acuerdo al modelo de referencia (las curvas están superpuestas), la señal de control I_B es imaginaria en ambos casos. Esto es debido a que la señal graficada es el cuadrado de la acción de control o equivalentemente el cuadrado del valor que toma la concentración de inductor del gen B (I_B^2) y al tomar valores negativos, cuando obtenemos la señal I_B aplicando la raíz cuadrada, el resultado es un número imaginario puro. En adición, la concentración de proteína A toma valores negativos en el ejemplo de $\lambda = -0{,}5$ para tiempos mayores de 2 horas. Por tanto, al tratarse todas las señales mencionadas de valores de concentraciones y tomar valores imaginarios y/o negativos, pierden su significado físico haciendo que el control sea irrealizable.

A expensas de un análisis posterior, sobre las regiones de validez de los parámetros y variables de nuestro control, que como hemos visto es necesario realizar para asegurarnos de que el sistema no pierde su significado físico. Es interesante comprobar como se comporta, bajo simulación, cuando una vez dentro de la variedad algebraica constituida por las soluciones del modelo de referencia, sacamos el sistema de la trayectoria, forzando una de las variables a tomar un valor que genere una solución fuera de la variedad y volvemos a aplicar la señal de control obtenida (29).

Utilizamos las mismas condiciones iniciales que en los ejemplos anteriores, descritas en las ecuaciones de equivalencia (31), para el caso particular de $\lambda = 1$, que como ya sabemos genera un comportamiento que mantiene el sentido físico del sistema.

En la figura (10) simulamos durante 10 horas el comportamiento del sistema, durante la 3 primeras horas, tramo de [0,3] Hr., no aplicamos la acción de control I_B y como podemos observar, en ese tramo, la concentración de la proteína B, no se comporta de acuerdo al modelo de referencia, lógico ya que no estamos controlando el sistema. Pasadas las 3 horas aplicamos la acción de control durante 3 horas más, tramo de [3,6] Hr., la acción de control sufre un desfase en forma de escalón y posteriormente sufre una singularidad que tratamos con más detalle en el siguiente

apartado, hasta que se estabiliza en el mismo valor que la concentración de inductor del gen A, I_A. Paralelamente la concentración de proteína B, tras el escalón, comienza a comportarse de acuerdo al modelo de referencia. Esto nos indica que el control, además de invariante es atractivo. Es decir, independientemente que las condiciones iniciales no pertenezcan a la variedad de las soluciones del modelo de referencia, al aplicar la señal de control y tras un transitorio, el sistema llega a la variedad objetivo. En el último tramo de cuatro horas, [6,10] Hr., anulamos de nuevo la señal de control, haciendo que el sistema salga de nuevo de la variedad y por tanto, la concentración de proteína B deja de seguir el modelo de referencia.

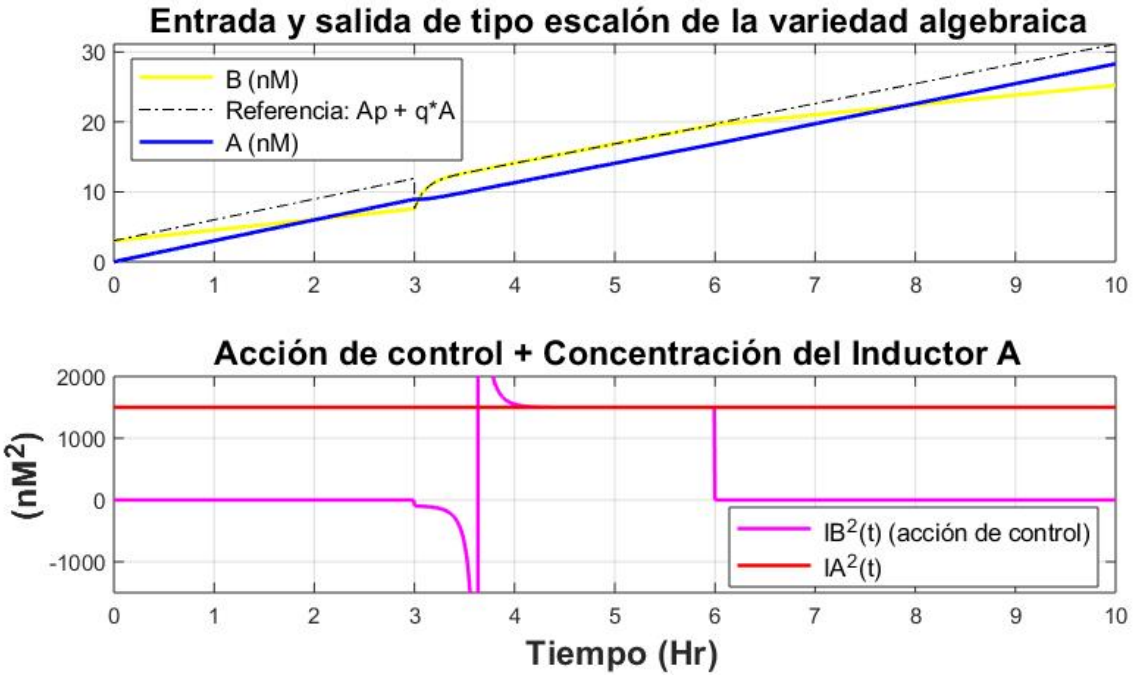


Figura 10: Simulación del efecto de la acción de control fuera y dentro de la variedad.

En principio, y olvidándonos de la respuesta de la señal de control de la figura (10), que analizamos con más detalle en el siguiente apartado, el control parece atractivo, esto nos indica que no sería necesaria una segunda capa de control a la hora de implementarlo.

Como ya sabemos, si bien el control es matemáticamente aplicable a todo valor que tomen las variables, éstas al ser concentraciones pierden su sentido físico fuera del rango de los número reales positivos. Además puesto que la señal de control (29) tiene estructura racional, conviene tener en cuenta algunas características de este tipo de expresiones como sus singularidades. Por ello es necesario estudiar los rangos de aplicabilidad del control, tanto para generar comportamientos físicamente imposibles como para evitar singularidades que impidan su implementación.

3.1.3. Análisis de resultados

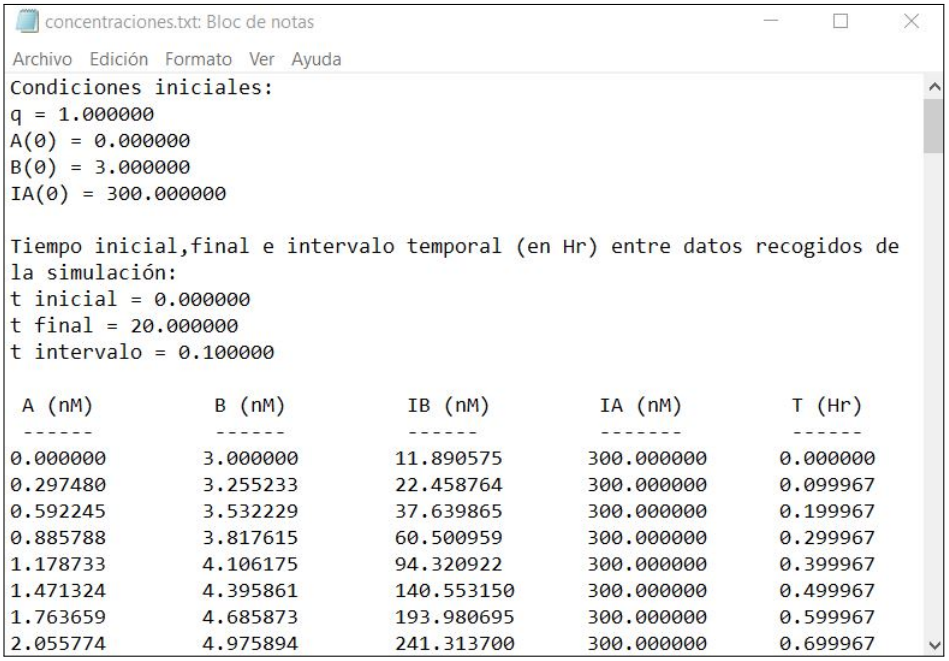
Si bien es necesario que en los controles intervengan magnitudes reales para poder ser implementables, no es el único requisito. En esta subsección analizaremos la viabilidad de la acción de control obtenida.

El primer punto a tener en cuenta es el rango de concentraciones en el que se desenvuelve un circuito genético como el de nuestro modelo. Si este rango no es realista, es decir, si las concentraciones son demasiado altas o bajas en comparación con el caso real, no será posible implementar el control.

En un circuito genético en el que participan dos genes, más concretamente en el caso de nuestro modelo, las concentraciones esperadas [12] son del orden de:

- Rango concentración de proteínas A y B entre: $[0, 100]$ nM.
- Rango de concentración de inductores I_A e I_B entre: $[0, 1000]$ nM

Para cercar los rangos de concentraciones que permiten un control físicamente implementable y realista, nos ayudamos de un *script* en *MATLAB* que nos permita extraer los datos de la simulación en formato de tabla para así escoger los valores de las concentraciones que mejor se adecuen a un caso realista. El procedimiento que hemos seguido es: a partir de las simulaciones físicamente realizables, como las de la figura (8), analizar que valores de concentraciones, a lo largo de la evolución del sistema se asemejan más al caso realista y simular de nuevo con usando los valores escogidos como nuevas condiciones iniciales. Sabemos que la efectividad del control estará garantizada por ser un control invariante y tratarse de un punto de la trayectoria.


concentraciones.txt: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
Condiciones iniciales:
q = 1.000000
A(0) = 0.000000
B(0) = 3.000000
IA(0) = 300.000000

Tiempo inicial,final e intervalo temporal (en Hr) entre datos recogidos de
la simulación:
t inicial = 0.000000
t final = 20.000000
t intervalo = 0.100000
```

A (nM)	B (nM)	IB (nM)	IA (nM)	T (Hr)
0.000000	3.000000	11.890575	300.000000	0.000000
0.297480	3.255233	22.458764	300.000000	0.099967
0.592245	3.532229	37.639865	300.000000	0.199967
0.885788	3.817615	60.500959	300.000000	0.299967
1.178733	4.106175	94.320922	300.000000	0.399967
1.471324	4.395861	140.553150	300.000000	0.499967
1.763659	4.685873	193.980695	300.000000	0.599967
2.055774	4.975894	241.313700	300.000000	0.699967

Figura 11: Tabla de valores, extraida con *script* a partir de la simulación, en formato .txt.

Los valores expresados en la tabla de la figura (11), provienen de la simulación, entre ellos, escogemos un conjunto de condiciones iniciales, correspondiente a valores que se dan en la realidad y que haga que nuestro sistema sea realista a la hora de ser implementado. Por ejemplo, escogemos el conjunto:

$$\begin{aligned} &A(0) = 7{,}8 \quad nM \\ &B(0) = 10{,}7 \quad nM \\ &I_A(0) = I_A(t) = 300 \quad nM \\ &\lambda = 1 \end{aligned} \tag{32}$$

y simulamos en la figura (12) su comportamiento para cerciorarnos de que es físicamente realizable.

Concluimos, tras observar la simulación del comportamiento del sistema en la figura (12), que la acción de control (29) actúa en el circuito genético de forma físicamente posible y realista, en el sentido que opera con valores de concentraciones reales y positivas, dentro de los rangos típicos, para todas y cada una de las variables que intervienen en control. La señal de control I_B es un curva suave sin singularidades, más adelante veremos la importancia de este hecho y la generación de proteína B sigue la dinámica deseada.

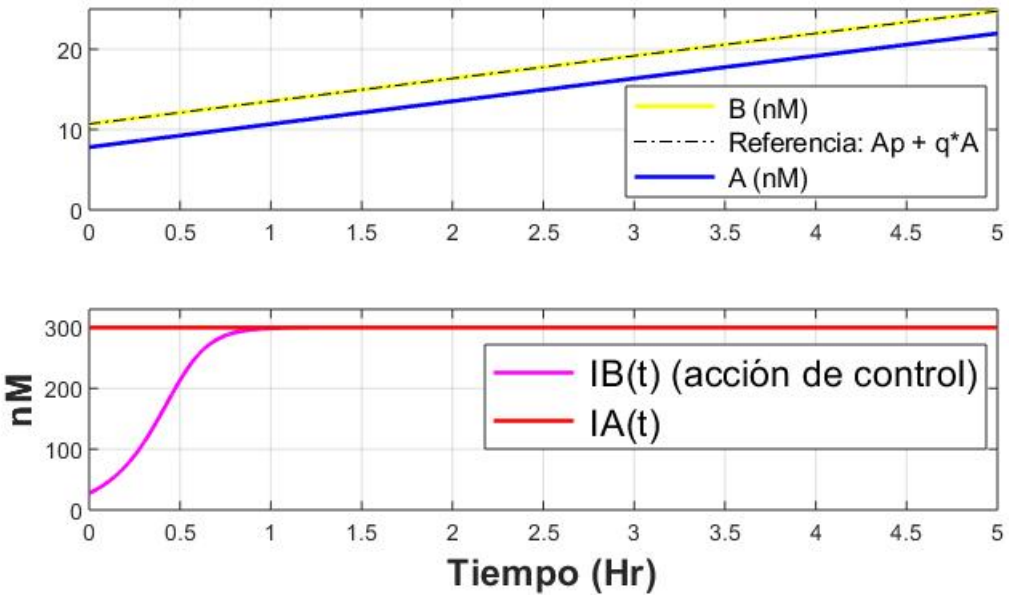


Figura 12: Simulación con condiciones realistas.

Una vez hemos determinado que la acción de control es viable, el siguiente punto del análisis es: si es posible implementarla en laboratorio. El hecho de que la acción de control sea un polinomio racional implica hacer hincapié en el estudio de sus propiedades, en concreto sus singularidades que generan un superficie a la que llamamos superficie de singularidad[15], para evitarla en laboratorio. Aunque según el método de implementación en laboratorio, este paso puede no ser necesario y solo debemos garantizar que cada uno de los monomios, o conjunto de monomios, en la expresión matemática de la acción de control, tiene su equivalente biológico. Aunque veremos este hecho, con más profundidad en el apartado de implementación.

Ya que en la expresión matemática de la acción de control (29) intervienen tres variables (A, B, I_A) y un parámetro (λ), para poder graficar la superficie de singularidad, es decir, la curva generada al anular el denominador, segmentamos la expresión graficando para distintos valores del parámetro λ. Así al tratarse de 3 variables y un valor concreto del parámetro proporcional λ, podemos observarla en 3 dimensiones. El hecho de tener una ayuda gráfica acerca de las regiones que producen una singularidad en la acción de control, nos ayuda a delimitar el rango en el que se deben mantener las variables.

Al tener, tanto el denominador como el denominador, ceros del mismo orden, podríamos pensar que la singularidad, en algunos casos, podría estar indeterminada y por tanto, tendríamos que estudiar la expresión mediante límites. Sin embargo, en este caso las superficies de singularidad del numerador y el denominador no se cruzan, es decir los ceros del sistema no coinciden para los mismos valores de concentraciones y por tanto, las singularidades vienen determinadas únicamente

[15]Superficie generada por los valores que anulan el denominador.

por los ceros del denominador.

En la siguiente figura (13) se muestra el hecho que las superficies formadas por los valores que anulan tanto el numerador (N) como el denominador (D), son paralelas. En la gráfica de perfil se aprecia mejor este hecho. Esto nos indica que no se anulan para los mismo valores y por tanto no hay indeterminaciones del tipo $\frac{0}{0}$, propagándose cada cero del numerador directamente a una singularidad de la acción de control.

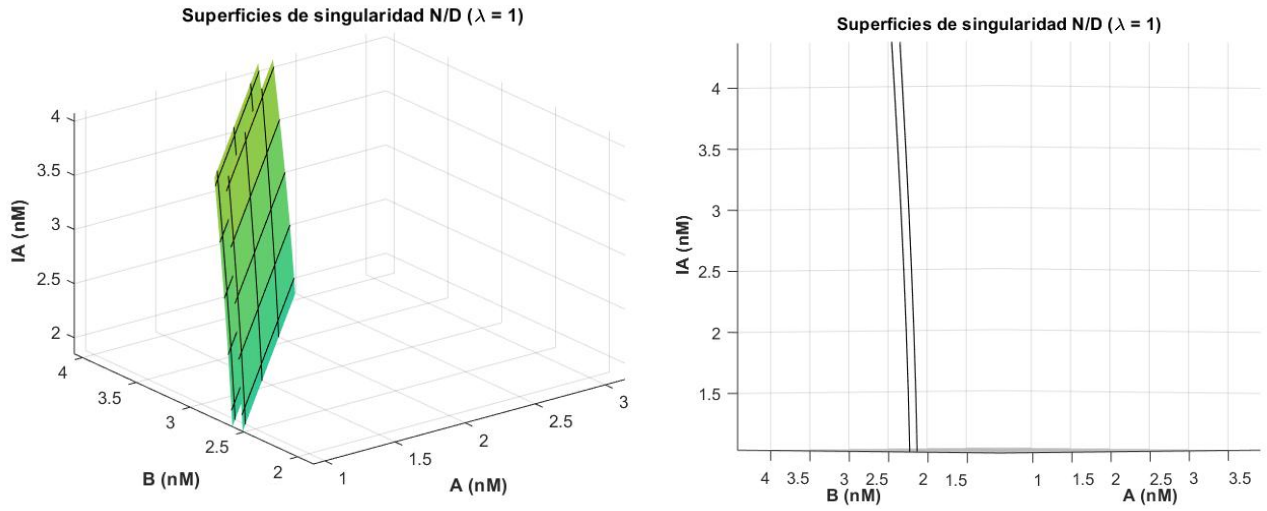


Figura 13: Comparativa entre superficies de singularidad N/D.

Aunque en la figura (13) solo se muestre el caso para $\lambda = 1$, el resultado es aplicable para todo valor del parámetro λ.

Con la seguridad de que la superficie de singularidad viene determinada por los valores que anulan el denominador, al representarla delimitamos la región de concentraciones que debemos evitar, tanto trabajar en ella como cruzarla, para que la señal de la acción de control tenga un comportamiento suave e implementable.

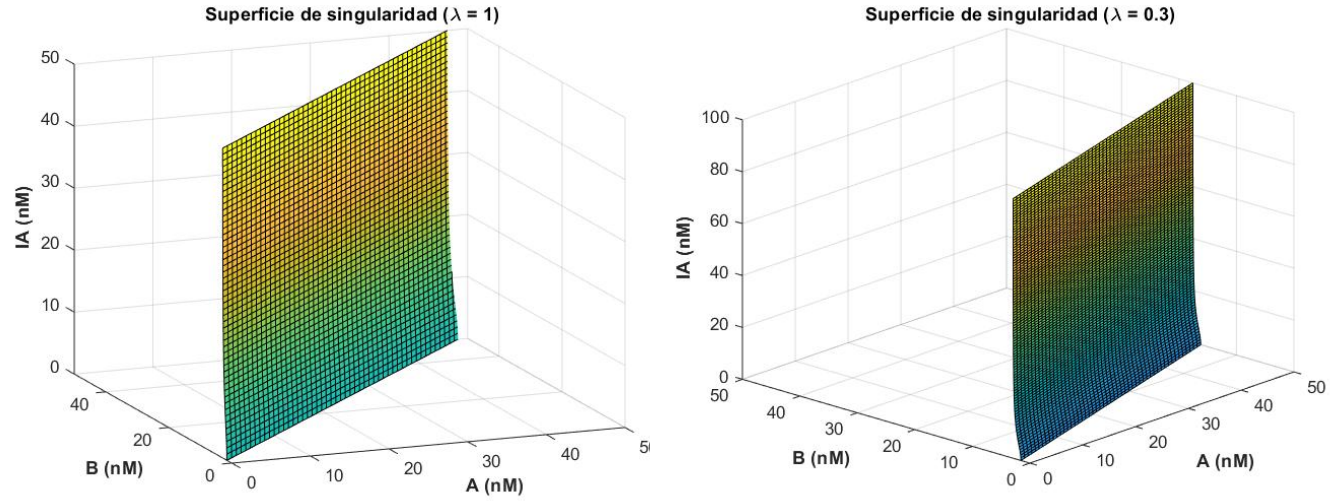


Figura 14: Superficie de singularidad, fragmentada, de la acción de control .

La primera característica que llama la atención de la superficie de singularidad representada en la figura anterior (14), es que según el valor del parámetro proporcional λ la superficie de singularidad cambia. Prestando más detalle, cuanto más cercano a 0 es el parámetro λ y por tanto más cercano a un derivador exacto el control, mayor es la inclinación de la curva tomando el eje de la proteína B como referencia. Esto quiere decir que, para mantener una acción de control suave, al no cruzar la superficie de singularidad, se requiere un aumento considerable de la concentración de proteína A en comparación con la de la proteína B, situación que para valores menores que $\lambda \sim 0{,}6$ no se puede dar debido a la propia dinámica del sistema. Otra característica a tener en cuenta es la forma de superficie vertical tomando como base los ejes de proteína A y B, esta característica nos dice que si nos encontramos en la región válida de concentraciones A y B prácticamente podemos variar la concentración de inductor del gen A sin miedo a que esta variación genere singularidades, ya que el desplazamiento es vertical y no cruzaría la superficie de singularidad. Pero como ya sabemos un incremento en el inductor del gen A genera un incremento en la concentración de proteína A pudiendo cruzar la superficie singularidad. De ahí la necesidad de buscar regiones de implementación como hemos hecho en el apartado anterior.

Vemos a que nos referimos, cuando hablamos de una singularidad en la acción de control, con el siguiente ejemplo. A partir de las condiciones iniciales,

$$\begin{aligned} &A(0) = B(0) = 3 \quad nM \\ &I_A^2(0) = 1500 \quad nM^2 \\ &\lambda = 0{,}3 \quad Hr^{-1} \end{aligned} \tag{33}$$

simulamos el comportamiento del sistema:

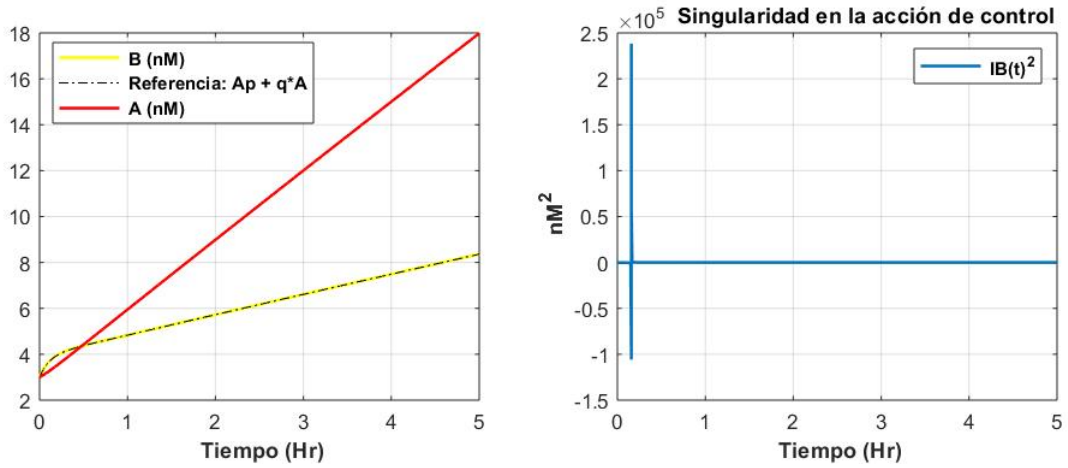


Figura 15: Ejemplo de singularidad en la acción de control.

En la figura (15) observamos que la respuesta del sistema es correcta, en términos de seguimiento del modelo de referencia y valores con sentido físico, sin embargo la acción de control al atravesar en algún momento la superficie de singularidad, pasa del lado negativo al positivo a desde la asíntota vertical, volviendo la señal de control inviable en términos de implementación.

En conclusión, mientras el control se aplique para un modelo de referencia no demasiado exigente $\lambda \gtrsim 0{,}6$ y dentro de región delimitada por la superficie de singularidad que genera una acción de control positiva, es decir una concentración del inductor I_B positiva, el control será **realista** e **implementable**.

3.1.4. Implementación en laboratorio

Como ya sabemos nuestro circuito genético se construye en el interior de una bacteria de *E.coli*. En ella introducimos los genes gA y gB y es la propia bacteria, mediante mecanismos internos, la que proporciona la polimerasa y los ribosomas. El control actúa variando la concentración de inductor, realimentado por el valor de las concentraciones de proteína.

El grupo de control de sistemas complejos de la UPV, trabaja con dos posibles formas de implementar el controlador obtenido para el circuito genético. Brevemente, la primera, a través de la medida de la concentración de proteínas (A, B), calcula la concentración necesaria de inductor (I_B) para que la dinámica del circuito siga la dinámica de referencia y la inyecta en el circuito, todo ello en tiempo real. Este método se conoce como implementación *in silico* [8]. La segunda, consiste en remplazar la acción de control por estructuras, como moléculas, que interaccionen con el circuito de forma que varíen su comportamiento del mismo modo que lo haría con la acción de control adecuada. Este método es conocido como implementación en células individuales (*single cell*).

3.1.4.1. In silico. O también hecho por computador.La implementación es llevada a cabo mediante un enfoque basado en sistemas de microfluídica, junto a una implementación por ordenador del algoritmo de control y un microscopio invertido que permite sensar la fluorescencia. A través de la siguiente imagen explicamos como contribuye cada uno de estos elementos en la implementación del control.

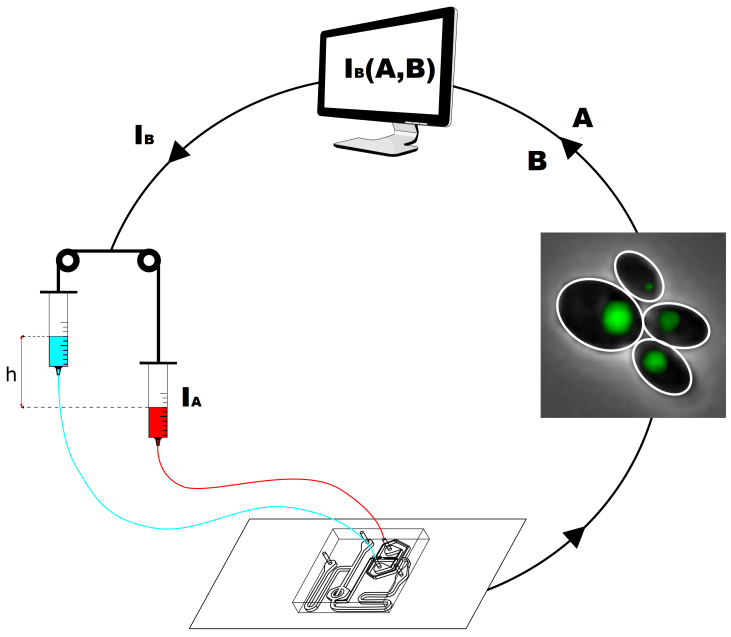


Figura 16: Implementación *in silico*.

Comenzamos por el microscopio invertido. El ordenador usa las imágenes tomadas por el microscopio para estimar la fluorescencia de las células y a partir de ésta medida, calcular la concentración de proteína A y B en la célula. Pero antes es necesario aplicar una técnica de espectroscopia basada en la absorbancia de la muestra de células, para determinar su número. Ésta consiste en iluminar la muestra con luz en torno a los $\sim 600\ nm$, correspondiente a la región del rojo visible. Dentro de ésta región del espectro electromagnético la absorbancia de la muestra un comportamiento exponencial conforme aumenta el número de células, que matemáticamente

corresponde a la expresión:

$$\dot{N}_{cells} = \mu \cdot N_{cells} \tag{34}$$

y a partir del correspondiente ajuste, con los datos experimentales, se obtiene el número de células de la muestra.

Llegados a este punto, conocemos el número de células, pero no la concentración de proteínas en cada una de ellas. Aquí entra en juego la técnica de medición basada en la fluorescencia. Existen un tipo de proteínas llamadas *reporter* que se adhieren al gen produciendo una proteína fluorescente por cada proteína que el gen produce. La interacción de estas proteínas con el resto de elementos del circuito es despreciable y por tanto no consideramos que influya en dinámica del mismo. Como existen proteínas *reporter* de varios colores, podemos asignar un color distinto a cada gen del circuito (gA, gB) y así medir cada concentración por separado.

La imagen tomada por el microscopio, recoge la fluorescencia de toda la muestra de células. Para estimar la concentración de proteína A y B en cada una de ellas, dividimos la fluorescencia total entre el número de células. Matemáticamente:

$$FOD = \frac{F}{OD} \tag{35}$$

donde las siglas OD hacen referencia a la densidad óptica, que es una magnitud de absorbancia, la F a fluorescencia y FOD al conjunto de ambas técnicas para estimar la concentración de cada tipo de proteína en cada una de las células.

A partir de la concentración de proteínas el ordenador calcula la señal de control necesaria I_B a partir de la acción de control obtenida en el desarrollo del proyecto (29). Una vez hecho es también el propio ordenador el que varía la altura entre las jeringas que contienen los inductores I_A e I_B, de esta forma la presión hidroestática generada por la relativa diferencia de alturas deriva en un flujo microfluidico hacia el dispositivo que contiene las células.

Por último, volvemos a medir la muestra, aproximadamente cada 15 minutos, comenzado el proceso de nuevo.

Veamos como, el método, estima un proceso relativamente lento en tiempo real. Supongamos que la proteína *reporter* es amarilla, llamada YFP por sus siglas en inglés. Usando un filtro de *Kalman*, que a partir de la información de las medidas más recientes de la FOD de YFP, estima los estados no medidos permitiendo la realimentación en tiempo real del algoritmo.

La principal desventaja de este método de implementación es que la célula ha de ser monitorizada durante todo el proceso de control. Pero por otra parte, si conseguimos que el control sea físicamente realizable, es decir, que las concentraciones sean reales y positivas dentro del rango de valores adecuado, como es nuestro caso, garantiza que el control sea **implementable**.

3.1.4.2. Single cell. O implementación en células individuales. Consiste en expresar la acción de control mediante estructuras conocidas como, por ejemplo, activadores o represores cuya función puede ser ejecutada, entre otros, por una molécula.

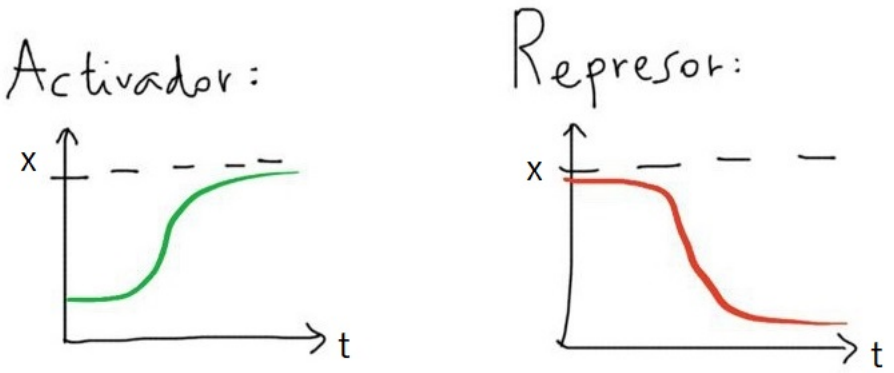


Figura 17: Ejemplo de activador y represor.

Todos los elementos del circuito genético son introducidos en un comienzo a través de jeringas. A partir de entonces la célula no requiere de intervención para realizar el control, como mucho podemos observar a través de técnicas FOD que el comportamiento es el adecuado. Son las propias estructuras, las que, elegidas adecuadamente, regulan el circuito de forma que se comporte de acuerdo a la dinámica de referencia. Estas estructuras se eligen a partir de la expresión de la acción de control (29). Por ejemplo, en caso de que identificaramos términos con la siguiente dependencia, en la propia expresión de la acción de control:

$$\frac{x^n}{x^n + c} \tag{36}$$

$$\frac{c_1}{x^n + c_2} \tag{37}$$

donde el término (36) corresponde a la estructura de un activador y el término (37) a la de un represor, siendo n el grado de cooperatividad, x la variable de concentración y c, c_1 y c_2 constantes. Podemos construir la acción de control a través de estructuras de este tipo, sustituyendo cada término por su estructura / molécula correspondiente, olvidándonos de La propia dinámica del sistema ya que ésta viene incluida en los propios elementos del circuito genético.

Sin embargo y para nuestro proyecto, este método hoy en día **no** es **implementable**. Es debido a la cantidad de monomios de la expresión matemática y a los signos negativos de algunos de ellos, usualmente asociados a degradaciones, que no puede expresarse totalmente en función de estructuras conocidas.

3.2. Control descartado

3.2.1. Modelo genético reducido

Si bien el control del modelo genético, tratado en este proyecto, es teóricamente implementable y realista, todavía no existen las técnicas experimentales para poder llevarlo a cabo en laboratorio con el método de *single cell*.

Esta sección surge para intentar, basándonos en el mismo modelo genético, ilustrar un ejemplo que pueda implementarse con estructuras biológicas experimentales actuales. Para ello hemos simplificado las ecuaciones del modelo genético (23), (24) y modificado ligeramente el significado del modelo de referencia (28), que sigue siendo el mismo.

La simplificación del modelo matemático surge al despreciar la interacción entre genes producida por la competición de recursos como los ribosomas (z) y la polimerasa (y) del medio:

$$J_A = J_B = 0$$

y reduciendo el coeficiente de *Hill*, reducimos el grado de cooperatividad entre el inductor y el factor de transcripción (TF), es decir reducimos el número de especies bioquímicas como el inductor necesarios para iniciar la transcripción:

$$n_A = n_B = 1$$

quedando definido el modelo matemático reducido, por las ecuaciones:

$$\dot{A} = \frac{K_{pA} \cdot K_{mA} \cdot c_A}{d_{mA} \cdot \nu_A \cdot K_A} \cdot z_T \cdot y_T \cdot f_A - d_A \cdot A \tag{38}$$

$$\dot{B} = \frac{K_{pB} \cdot K_{mB} \cdot c_B}{d_{mB} \cdot \nu_B \cdot K_B} \cdot z_T \cdot y_T \cdot f_B - d_B \cdot B \tag{39}$$

Donde:

$$\begin{aligned} f_A &= \frac{\frac{K_A}{K_A^0} + \frac{1}{K_{nA}} \cdot I_A}{1 + \frac{1}{K_{nA}} \cdot I_A} \\ f_B &= \frac{\frac{K_B}{K_B^0} + \frac{1}{K_{nB}} \cdot I_B}{1 + \frac{1}{K_{nB}} \cdot I_B} \end{aligned} \tag{40}$$

son las funciones de *Hill*, un tipo de función sigmoide que describe las interacciones con el inductor.

3.2.1.1. Variaciones en el diseño del control. Como hemos comentado en el apartado anterior, si bien la dinámica de referencia es la misma que en el modelo completo ésta adquiere un significado diferente debido a los nuevos requisitos de diseño:

- Una concentración del inductor del gen B (I_B) cualquiera a lo largo del tiempo.
- Construimos la acción de control sobre la concentración de inductor del gen A I_A con dependencia explícita de las dos proteínas $I_A = I_A(A, B)$
- Constantes de traducción Kp_A, Kp_B como parámetros libres.
- Mismo modelo de referencia que en el modelo completo: $B(t) = \dot{A}(t) + \lambda \cdot A(t)$

Al ser, en este caso, la concentración del inductor del gen A, I_A la entrada de control del sistema y como ésta afecta directamente a la sintetización de proteína A, la dinámica de referencia (28) corresponde a un integrador, en que la sintetización de proteína A es proporcional a la integral de la concentración de proteína B más un término. A diferencia del modelo completo donde cumplía un papel de derivador.

3.2.1.2. Expresión de la señal de control. Al igual que en el modelo completo aplicamos el algoritmo de *Rosenfeld-Groebner* al conjunto de polinomios formado por el modelo matemático (38), (39) y el modelo de referencia (28). Usamos un ranking de eliminación que triangularice el sistema de ecuaciones dejando la concentración del inductor en función del resto de variables y parámetros y, como indica uno de los requisitos de diseño, dejamos las constantes de traducción como parámetros libres. Esta configuración queda reflejada en el anillo diferencial:

$R :=$ *DifferentialRing*(*blocks*=[IA,[A,B,IB]], *derivations*=[t], *arbitrary*=[λ, Kp_A, Kp_B]):

y tras aplicar el algoritmo de *Rosenfeld-Groebner* al conjunto de polinomios, analizar las expresiones obtenidas candidatas a acción de control y verificar que son capaces de reproducir la referencia, obtenemos la siguiente expresión:

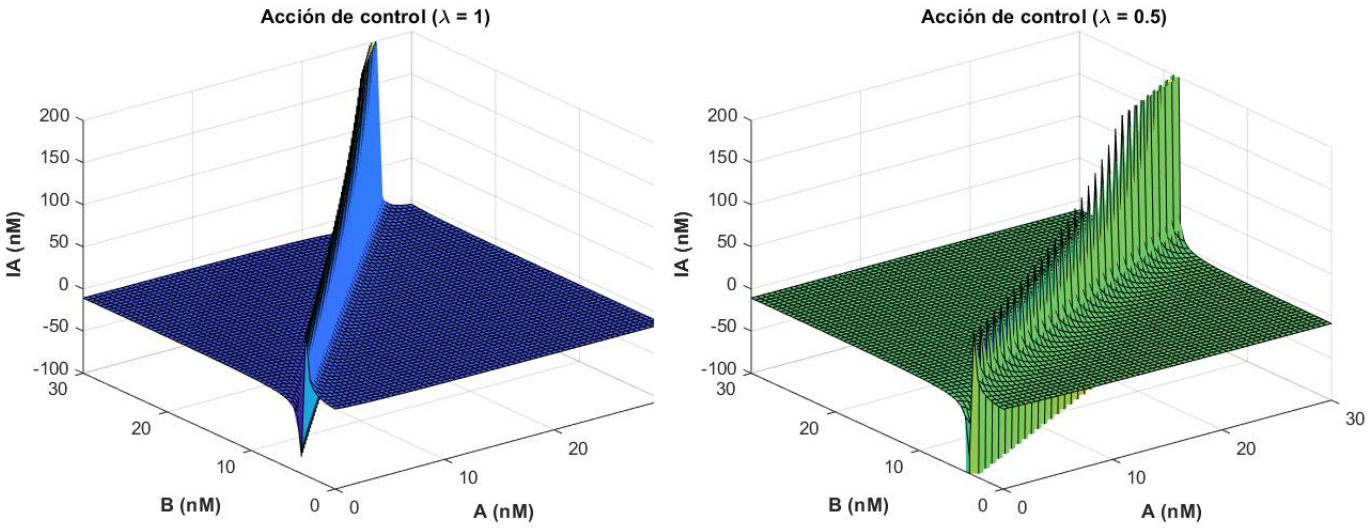
$$I_A(t) = \frac{-10 \cdot (-150 \cdot \lambda \cdot A(t) - A(t) - 150 \cdot B(t) + 168 \cdot Kp_A)}{-150 \cdot \lambda \cdot A(t) - A(t) - 150 \cdot B(t) + 336 \cdot Kp_A} \tag{41}$$

A primera vista llama la atención dos aspectos de la ecuación (41), la primera es que la expresión del numerador es prácticamente proporcional a la expresión del denominador, a excepción del termino independiente. Esto genera, al igual que en el caso del modelo original, que las singularidades vengan determinadas directamente por los ceros del denominador.

El segundo aspecto es la ausencia de la parte correspondiente al gen B en el modelo, indicándonos que al despreciar la interacción entre ellos ($J_A = J_B = 0$) la dinámica del gen B no influye en el comportamiento del gen A. Este resultado tiene una interpretación potencialmente interesante debido a que, con la misma señal de control (41) podemos usar como referencia cualquier proteína que no interaccione con el modelo.

3.2.1.3. Simulación y análisis de resultados. Al estar definida la acción de control (41) por dos variables (A y B) y el parámetro de proporcionalidad del modelo de referencia (λ), podemos simular la curva de respuesta que genera I_A para cualquier valor de concentración de proteínas, segmentada para distintos valores del parámetro λ.

Cogiendo el valor de Kp_A de la tabla (1), de forma orientativa. Graficamos la acción de control I_A:



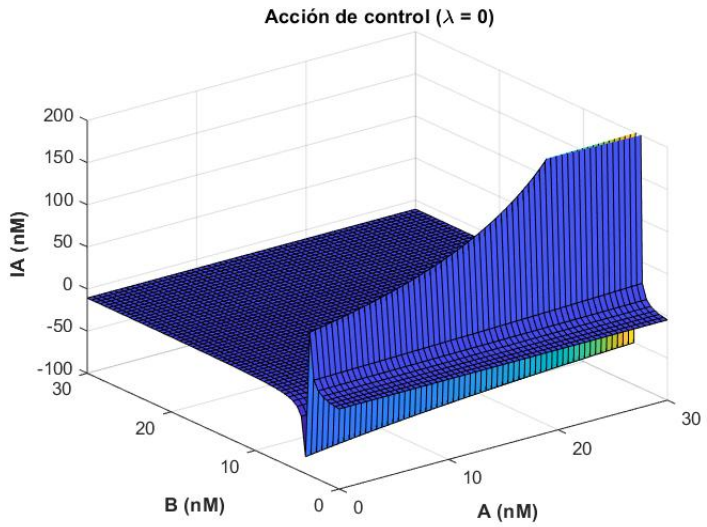


Figura 18: Superficie de la acción de control, para distintos valores del parámetro proporcional λ.

Al igual que en modelo original, la singularidad es vertical al plano concentración de A - concentración de B y además ésta aumenta su ángulo con respecto al eje de la concentración de proteína B conforme disminuye el parámetro λ llegando a ser totalmente perpendicular cuando $\lambda = 0$.

Como ya hemos discutido la dinámica del gen B no influye en la dinámica del integrador, esto es equivalente a poder integrar la concentración de cualquier proteína independiente del circuito. En nuestro caso y por simplificar, escogemos un valor de concentración constante a lo largo del tiempo para la proteína B además de ejecutar un integrador exacto ($\lambda = 0$), por lo que el modelo de referencia queda como:

$$A(t) = \int B(t)dt \tag{42}$$

y como ya sabemos del modelo original, tenemos que trabajar sobre una región donde todos los valores de las variables sean reales y positivos, es decir que mantengan el significado físico. En esta ocasión como si es posible graficar la superficie de la acción de control para $\lambda = 0$, es posible identificar gráficamente la región de validez de control:

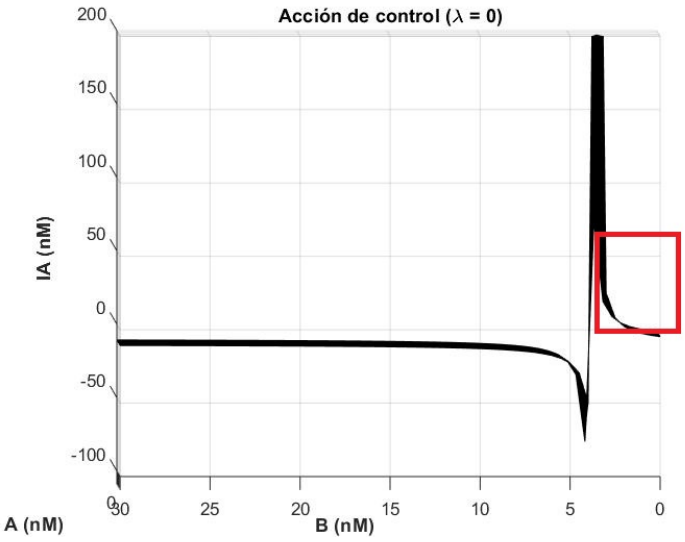


Figura 19: Región de trabajo para que el control sea físicamente implementable.

que como vemos en la figura (19) está delimitada para una concentración de la proteína B entre $\sim [0, 4]\ nM$ y una concentración de proteína A entre $\sim [0, \infty)$.

Con la ayuda de *SIMULINK* simulamos la dinámica del controlador, construido a partir de la ecuación (38) y la acción de control (41).

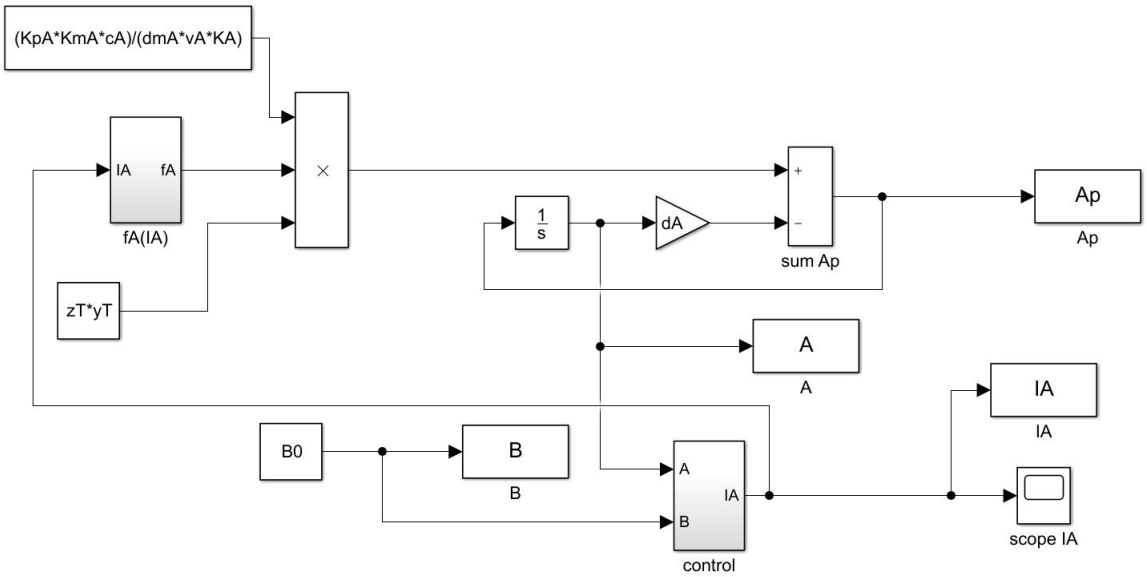


Figura 20: Esquema de *SIMULINK* del modelo reducido.

Como vemos en el esquema de la figura (20) anterior y como habíamos adelantado, la concentración de proteína B viene dada por un bloque constante, por lo que al tratarse de un integrador puro $\lambda = 0$ esperamos un comportamiento lineal creciente con el tiempo de la proteína A. Para los valores iniciales: $A(0) = 1$, $B(t) = B(0) = 3$ obtenemos el siguiente comportamiento:

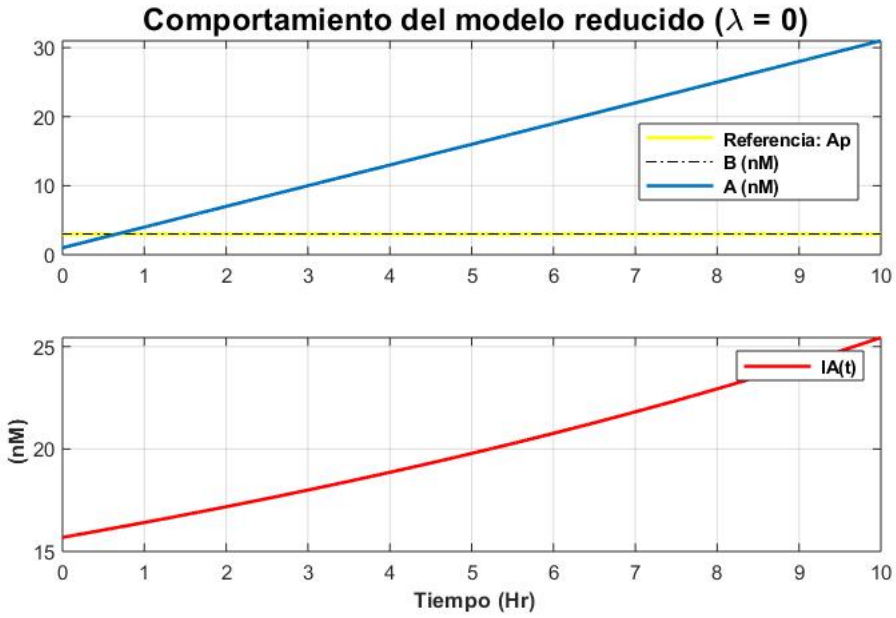


Figura 21: Comportamiento del sistema reducido + acción de control.

y efectivamente, como esperábamos, la concentración de proteína A es lineal con el tiempo con una pendiente de $3\frac{nM}{Hr}$, hecho que confirma que el integrador funciona correctamente al ser todas

las variables reales y positivas. El inconveniente llega cuando analizamos la acción de control para periodos de tiempo más amplios, ya que en principio no parece adoptar una sigmoidea que se estabiliza en un valor constante, como en el resto de casos.

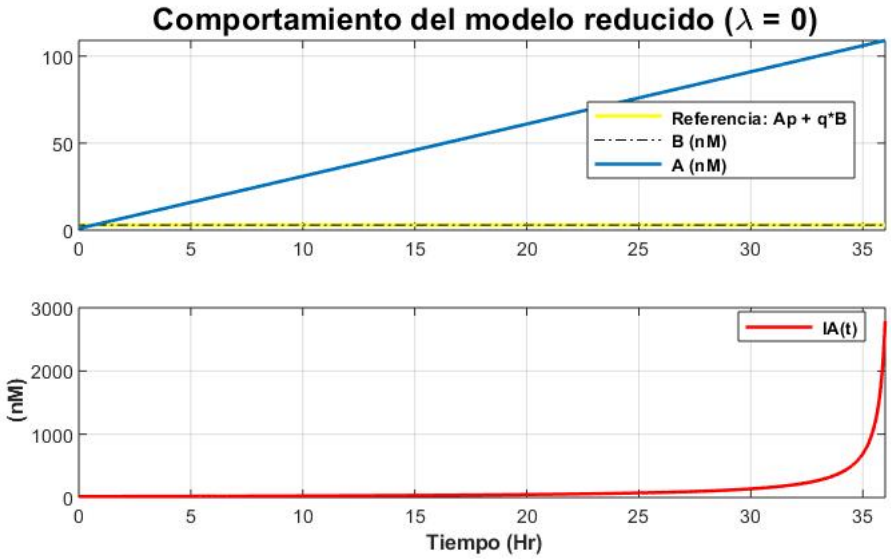


Figura 22: Comportamiento del sistema reducido + acción de control.

Simulando de nuevo (ver figura 22) comprobamos que control no es físicamente realizable para todo instante de tiempo, ya que en algún momento abandona la región de trabajo que hemos definido en la figura (19). Esto es debido a la proximidad con la singularidad a la que estamos obligados a trabajar. La forma de solucionar este problema requiere de métodos matemáticos fuera del área de estudio de este proyecto que puedan aproximar la acción de control (41), en la región de trabajo definida en (19) por una expresión sin singularidades o ha restringirse a casos particulares, tanto del modelo de referencia, como de las condiciones iniciales que, a lo largo del tiempo, no abandonen la región de trabajo.

Ahora que sabemos que la acción de control necesita de un estudio más profundo a la hora de implementarse con garantías. En el hipotético caso de evitar la singularidad, el último paso es conseguir expresar la acción de control mediante estructuras matemáticas conocidas, (en el sentido que son identificables con el comportamiento de, por ejemplo, una molécula) para su implementación en laboratorio.

3.2.1.4. Implementación en células individuales. En inglés *single cell*, pretende mediante el propio circuito genético diseñado con genes y las reacciones químicas que tengan lugar entre ellos, ejecutar la acción de control dentro de la célula. Es decir, que este controlador está restringido a las propias interacciones entre genes y a la estructura del circuito.
Trabajando matemáticamente la expresión de la señal de control (41) llegamos a la expresión:

$$I_A = -10 + \frac{1680 \cdot Kp_A}{150 \cdot \lambda \cdot A - A - 150 \cdot B + 336 \cdot Kp_A} \tag{43}$$

y para identificar la estructura renombramos las variables: $x = 150\lambda A - A - 150B$, $c = 336Kp_A$, obteniendo:

$$I_A = -10 + \frac{5c}{x + c} \tag{44}$$

que como vimos en la figura (17), recuerda a una función de *Hill* de un represor. Sin embargo, aunque hoy en día este tipo de estructuras son conocidas e incluso algunas implementables, la relación estática entre los genes gA y gB descrita en la variable x de la ecuación (44) genera problemas experimentales que hoy en día no están resueltos, pero es posible que en un futuro cercano y gracias a nuevas tecnologías si lo estén.

Concluimos remarcando que, aunque la implementación en *single cell* con las técnicas actuales **no es posible**, la estructura matemática de la acción de control (44) plantea buenas expectativas de serlo en un futuro próximo.

4. Conclusión

Concluimos el proyecto resaltando, de forma sintetizada, cada uno de los puntos de interés tratados a lo largo del trabajo.

En cuanto al método de diseño de controles invariantes: el algoritmo de *Rosenfeld-Groebner* es un herramienta del álgebra computacional, implementada dentro del *software* comercial *Maple*. Esto permite que cualquier persona tenga acceso a su uso. Su aplicación abarca un espectro amplio de sistemas no lineales, en concreto aquellos cuya dinámica es modelable a través de ecuaciones diferenciales polinomiales y/o algebraicas.

Acerca del controlador obtenido. En primer lugar, éste es expresado matemáticamente de forma simbólica. Lo que permite entre otras cosas, comprender y analizar con más detalle la acción del control e incluso trabajar con la expresión de control, por ejemplo simplificándola mediante aproximaciones matemáticas, siempre bajo justificación. Al ser el control invariante, sabemos que, bajo un modelo ideal y sin perturbaciones, una vez apliquemos el control de forma efectiva, éste lo será para todo instante de tiempo. En el caso de que la variedad objetivo no fuese atractiva, la técnica de diseño sigue teniendo sentido práctico, ya que podemos diseñar una segunda capa de control que la vuelva atractiva y por tanto, el control invariante una vez atraido dentro de la variedad, actuaría correctamente.

En segundo lugar, recordar que la acción de control obtenida es físicamente implementable en laboratorio. Actualmente es posible implementarla *in silico* que, haciendo memoria y de forma breve, consiste en calcular la señal de control necesaria en cada instante mediante computador y consecuentemente actuar sobre el sistema a tiempo real. Los resultados obtenidos acerca del método de implementación en células individuales han incentivado un futuro estudio, siguiendo la linea de este proyecto, para intentar llevarlo a cabo. Esto es debido a que en el circuito genético, al modelarse las reacciones bioquímicas a partir de ecuaciones diferenciales polinomiales, la técnica del proyecto permite estudiar todo tipo de configuraciones y de comportamientos objetivo, e incluso reescribir el sistema de ecuaciones en términos más adecuados para su implementación en células individuales.

En general, es un método de diseño de controles invariantes con resultados probados en el campo de la biología sintética, que requiere de una investigación más profunda para explotar totalmente los beneficios que puede aportar a los sistemas de control y que es extrapolable a todo tipo de sistemas no lineales cuyo modelo matemático cumpla los requisitos de expresión en términos polinomiales.

Bibliografía

[1] Alberts, B., Johnson, A., Lewis, J., Raff, M., Roberts, K., and Walter, P. Molecular biology of the cell (garland, new york, 2002). *Google Scholar*.

[2] Alon, U. *An introduction to systems biology: design principles of biological circuits*. Chapman and Hall/CRC, 2006.

[3] Boada, Y., Vignoni, A., and Picó, J. Engineered control of genetic variability reveals interplay among quorum sensing, feedback regulation, and biochemical noise. *ACS synthetic biology 6*, 10 (2017), 1903–1912.

[4] Boada, Y., Vignoni, A., and Pico, J. Promoter and transcription factor dynamics tune protein mean and noise strength in a quorum sensing-based feedback synthetic circuit. *bioRxiv* (2017), 106229.

[5] Cox, D., Little, J., and O'shea, D. *Ideals, varieties, and algorithms*, vol. 3. Springer, 2007.

[6] Dominguez, S., Campoy, P., and Sebastián, J. M. *Control en el espacio de estado*. Universidad Politécnica de Madrid, Escuela Técnica Superior de Ingenieros Industriales, 2000.

[7] Krstic, M., Kanellakopoulos, I., Kokotovic, P. V., et al. *Nonlinear and adaptive control design*, vol. 222. Wiley New York, 1995.

[8] Menolascina, F., Fiore, G., Orabona, E., De Stefano, L., Ferry, M., Hasty, J., di Bernardo, M., and di Bernardo, D. In-vivo real-time control of protein expression from endogenous and synthetic gene networks. *PLoS computational biology 10*, 5 (2014), e1003625.

[9] Ogata, K. *Ingeniería de control moderna*. Pearson Educación, 2003.

[10] Picó, J., Vignoni, A., Picó-Marco, E., and Boada, Y. Modelado de sistemas bioquímicos: De la ley de acción de masas a la aproximación lineal del ruido. *Revista Iberoamericana de Automática e Informática Industrial RIAI 12*, 3 (2015), 241–252.

[11] Pico-Marco, E. Differential algebra for control systems design: constructive computation of canonical forms. *IEEE Control Systems 33*, 2 (2013), 52–62.

[12] Qian, Y., Huang, H.-H., Jiménez, J. I., and Del Vecchio, D. Resource competition shapes the response of genetic circuits. *ACS synthetic biology 6*, 7 (2017), 1263–1272.

[13] Ritt, J. F. *Differential equations from the algebraic standpoint*, vol. 14. American Mathematical Soc., 1932.

[14] Zhang, C., Tsoi, R., and You, L. Addressing biological uncertainties in engineering gene circuits. *Integrative Biology 8*, 4 (2016), 456–464.

Apéndice

Conceptos de álgebra abstracta

Grupo

Estructura algebraica formada por el par $(G, *)$; donde G es un conjunto cualquiera y $*$ una ley de composición interna [16] que cumple las propiedades siguientes:

I) Asociativa: $x * (y * z) = (x * y) * z \quad \forall x, y, z \in G$

II) Elemento neutro: $\exists\ e \in tal\ que\ \forall x \in G, e * x = x * e = x$

III) Elemento simétrico: $\forall x \in G, \exists\ x^{-1} \in G\ tal\ que\ x * x^{-1} = x^{-1} * x = e$

Si además, la ley $*$ es conmutativa, el grupo se llama **abeliano**.

Anillo

Sea A un conjunto donde hemos definido dos operaciones o leyes de composición internas $(\perp)$ y $(*)$. La estructura algebraica $(A; \perp, *)$ es un anillo si se satisface las condiciones siguientes:

I) $(A, \perp)$ es un grupo abeliano.

II) La ley $*$ es asociativa: $(x * y) * z = x * (y * z)$.

III) La ley $*$ es distributiva respecto de la ley $\perp$:

$$x * (y \perp z) = (x * y) \perp (x * z) \qquad (x \perp y) * z = (x * z) \perp (y * z)$$

Si existe el elemento neutro para la segunda ley $(*)$, el anillo se llama unitario.

Si la segunda ley $(*)$ es conmutativa, el anillo se llama conmutativo o abeliano.

Cuerpo

Sea K un conjunto donde hemos definido dos operaciones o leyes de composición internas ($\perp$, usualmente aditiva) y $(*)$, usualmente multiplicativa). La estructura algebraica $(K; \perp, *)$ es un cuerpo si se satisfacen las condiciones siguientes:

I) $(K, \perp)$ es grupo abeliano.

II) $(K - \{0\}, *)$ es grupo abeliano.

III) La ley $*$ es distributiva respecto a la ley $\perp$.

El conjunto de los números reales $(\Re)$ con las operaciones de suma y producto usuales es un ejemplo de cuerpo.

[16] Una ley de composición interna asigna a cada par ordenado (a, b), cuyas componentes pertenecen ambas al conjunto A, un tercer elemento c, también contenido en A.

Ideal

Un subconjunto I perteneciente al anillo A, es un ideal si satisface:

I) $0 \in I$

II) $\forall f, g \in I : f + g \in I$.

III) $h \in A \wedge f \in I : h \cdot f \in I$.

En consecuencia, un conjunto de polinomios $f_1, ..., f_s$ perteneciente al anillo polinomial[17] $K[x_1, ..., x_n]$ generan un ideal que denotamos como:

$$\langle f_1, ..., f_s \rangle = \{\sum_{i=1}^{s} h_i f_i : h_1, ..., h_s \in K[x_1, ...x_n]\}$$

Llamamos **radical** del ideal, denotado como $\sqrt{I}$, al conjunto que cumple:

$$f : f^m \in I \ ; m \geq 1$$

que, por definición; $I \subset \sqrt{I}$

Base de un espacio vectorial

Llamamos base de un espacio vectorial [18], E, a cualquier conjunto de vectores, $B = \{|e_i\rangle\}$ que satisfaga las condiciones siguientes:

1. B es un sistema de vectores linealmente independientes.
2. Cualquier vector $|v\rangle \in E$ se puede escribir como una combinación lineal de los vectores de B:
$$|v\rangle = \sum_i \alpha_i \, |e_i\rangle$$

[17] Anillo de polinomios sobre un cuerpo (recordatorio: un anillo no tiene elemento simétrico con la operación $*$).
[18] Llamamos espacio vectorial a la estructura algebraica formada por la terna $\{(E, +), (K, +, \cdot), (\cdot)\}$.

Programas modelo genético

Programa Maple

GENETIC CIRCUITS
Parámetros:

$z_T := 140$:
$y_T := 200$:

>Circuito A:

$Kp_A := 100/60$:
$Km_A := 100/60$:
$cA := 60$:
$d_{mA} := 10/60$:
$d_A := 4/600$:
$n_A := 2$:
$K_{0A} := 1000$:
$K_A := 500$:
$K_{nA} := 10^{nA}$:
$\nu_A := 15000$:
$J_A := \frac{cA}{K_A}\left(1 + \frac{k_{mA}\cdot y_T}{d_{mA}\cdot \nu_A}\right)$:

>Circuito B:

$Kp_B := 100/60$:
$Km_B := 100/60$:
$cB := 60$:
$d_{mB} := 10/60$:
$d_B := 4/600$:
$n_B := 2$:
$K_{0B} := 1000$:
$K_B := 500$:
$K_{nB} := 10^{nB}$:
$\nu_B := 15000$:
$J_B := \frac{cB}{K_B}\left(1 + \frac{k_{mB}\cdot y_T}{d_{mB}\cdot \nu_B}\right)$:

>Sistema de Ecuaciones del modelo:

$f_A := \frac{\frac{K_A}{K_A^0} + \frac{1}{K_{nA}}\cdot I_A^{nA}}{1 + \frac{1}{K_{nA}}\cdot I_A^{nA}}$:

$f_B := \frac{\frac{K_B}{K_B^0} + \frac{1}{K_{nB}}\cdot I_B^{nB}}{1 + \frac{1}{K_{nB}}\cdot I_B^{nB}}$:

$z \cdot y := \frac{z_T \cdot y_T}{1 + J_A \cdot f_A + J_B \cdot F_B}$:

$p1 := \frac{K_{pA} \cdot K_{mA} \cdot c_A}{d_{mA} \cdot \nu_A \cdot K_A} \cdot z \cdot y \cdot f_A - d_A = diff(A(t), t) :$

$p2 := \frac{K_{pB} \cdot K_{mB} \cdot c_B}{d_{mB} \cdot \nu_B \cdot K_B} \cdot z \cdot y \cdot f_B - d_B = diff(B(t), t) :$

>Modelo de referencia:

$ref := diff(A(t), t) + q \cdot A(t) = B(t) :$

>Conjunto de polinomios sobre el que aplicamos el algoritmo de Rosenfeld-Groebner:

$P := [p1, p2, ref] :$

>Llamamos a la librería que nos permite utilizar herramientas de álgebra diferencial:

$with(DifferentialAlgebra) :$

>Definimos el anillo sobre el que actuará el algoritmo de R-G. El apartado block hace referencia al ranking empleado, en este caso hemos escogido el más óptimo computacionalmente para Maple usando doble corchete ([[...]]). Las derivadas son temporales y hay un parámetro ajustable procediente del modelo de referencia (q):

$R := DifferentialRing(blocks = [[IB, A, IA, B]], derivations = [t], arbitrary = [q]) :$

>Obtenemos la base de Groebner del conjunto de polinomios P y escogemos, entre uno de los ideales que la forman, una expresión que nos sirva como futura acción de control, por ejemplo:

$simplify(Equations(G[1], solved)[3])$

$$I_B(t)^2 = -50\frac{(-45150A(t)I_A(t)^2q+45150B(t)I_A(t)^2+301A(t)I_A(t)^2-4260000qA(t)-140000I_A(t)^2+4260000B(t)+28400A(t)-7000000)}{-23850A(t)I_A(t)^2q+23850B(t)I_A(t)^2+159A(t)I_A(t)^2-2257500qA(t)-70000I_A(t)^2+2257500B(t)+15050A(t)-3500000}$$

>Ahora comprobamos, de nuevo mediante el algoritmo de Rosenfeld-Groebner, si la acción de control elegida es capaz de reproducir el modelo de referencia:

$PC := [p1, p2, Equations(G[1], solved)[3]] :$
$GC := RosenfeldGroebner(PC, R) :$
$BelongsTo(ref, GC)$

$true$

>Al pertenecer el modelo de referencia (ref) a la base de Groebner, obtenida a partir de los polinomios del circuito genético ($p1$, $p2$) y la expresión de la acción de control ($I_B(t)^2$), podemos concluir que el control es adecuado.

Programa MATLAB

```
%% CIRCUITO GENETICO

A_0 = 6.41; % Concentracion inicial de proteina A (nM)
B_0 = 9.23; % Concentracion inicial de proteina B (nM)
IA2_0 = 1500; % Concentracion al cuadrado de Inductor del gen A
T = 10; %En horas

q=1; %parametro proporcional del Modelo de referencia B=Ap + q*A

sim('genes_ref2.slx'); % Ejecuta el SIMULINK del modelo

%% GRAFICAS

figure
subplot(211)
plot(Ap.time,Ap.signals.values + q*A.signals.values,'LineWidth',1.5)
grid on, hold on
plot(B.time,B.signals.values,'LineWidth',1.5)


%Anadimos la senal de A(t)
plot(A.time,A.signals.values,'LineWidth',1.5)
legend('Referencia: Ap + q*A','B(nM)','A(nM)')
title('\fontsize{14} Concentracion de proteinas A, B + Modelo de
    referencia')

subplot(212)

%Raiz de la senal IB(t)
plot(IB2.time,sqrt(IB2.signals.values),'LineWidth',1.5)
ylabel('\bf \fontsize{14} (nM)'),grid on

%Raiz de la senal IA(t)
hold on
plot(IA2.time,sqrt(IA2.signals.values),'LineWidth',1.5)
legend('\fontsize{14} IB(t) (accion de control)','\fontsize{14} IA(t)'
    )
xlabel('\bf \fontsize{14} Tiempo (Hr)')\
title('\fontsize{14} Accion de control + Concentracion del Inductor A'
    )

%% BLOC DE NOTAS (archivo .txt)

%Genera un archivo .txt con los datos de la simulacion distribuidos
    en columnas

text = fopen('concentraciones.txt','wt');

```

```
45 %Tiempos de extracion de datos
46 t_i=0;
47 t_f=10;
48 t_s=0.1; %Intervalo de tiempo entre datos
49
50 fprintf(text,'Condiciones iniciales:\n');
51 fprintf(text,'q = %f',q);
52 fprintf(text,'\nA(0) = %f',A_0);
53 fprintf(text,'\nB(0) = %f',B_0);
54 fprintf(text,'\nIA(0) = %f',sqrt(IA2_0));
55
56 fprintf(text,'\n\nTiempo inicial, final e intervalo temporal (en Hr)
     entre datos recogidos de la simulacion:\n')
57
58 fprintf(text,'t inicial = %f',t_i);
59 fprintf(text,'\nt final = %f',t_f);
60 fprintf(text,'\nt intervalo = %f',t_s);
61
62 fprintf(text,'\n\n A (nM) \t B (nM) \t IB (nM) \t IA (nM) \t T (Hr)\n'
     );
63 fprintf(text,' ------ \t ------ \t ------ \t ------- \t ------');
64
65 for i=t_i*10000:t_s*10000:t_f*10000
66     fprintf(text,'\n%f',A.signals.values(i+1));
67     fprintf(text,'\t%f',B.signals.values(i+1));
68     fprintf(text,'\t%f',sqrt(IB2.signals.values(i+1)));
69     fprintf(text,'\t%f',sqrt(IA2.signals.values(i+1)));
70     fprintf(text,'\t%f',A.time(i+1));
71 end
72 winopen('concentraciones.txt');
```

Programas modelo genético reducido

Programa Maple

GENETIC CIRCUITS
Parámetros:

$z_T := 140$:
$y_T := 200$:

>Circuito A:

$Km_A := 100/60$:
$cA := 60$:
$d_{mA} := 10/60$:
$d_A := 4/600$:
$n_A := 1$:
$K_{0A} := 1000$:
$K_A := 500$:
$K_{nA} := 10^{nA}$:
$\nu_A := 15000$:
$J_A := 0$:

>Eliminamos la dinámica del gen gB para comprobar que obtenemos el mismo resultado.

>Sistema de Ecuaciones del modelo:

$f_A := \frac{\frac{K_A}{K_A^0} + \frac{1}{K_{nA}} \cdot I_A^{nA}}{1 + \frac{1}{K_{nA}} \cdot I_A^{nA}}$:

>El comando simplify() simplifica la expresión simbólicamente, bien de forma algebraica o sustituyendo valores ya declarados.

$simplify(f_A)$

$\frac{5 + I_A(t)}{10 + I_A(t)}$

$z \cdot y := \frac{z_T \cdot y_T}{1 + J_A \cdot f_A + J_B \cdot F_B}$:

$simplify(f_A)$

28000

$p1 := \frac{K_{pA} \cdot K_{mA} \cdot c_A}{d_{mA} \cdot \nu_A \cdot K_A} \cdot z \cdot y \cdot f_A - d_A = diff(A(t), t)$:

>Modelo de referencia:

$ref := diff(A(t), t) + q \cdot A(t) = B(t)$:

>Conjunto de polinomios sobre el que aplicamos el algoritmo de Rosenfeld-Groebner:

$P := [p1, ref]$:

>Llamamos a la librería que nos permite utilizar herramientas de álgebra diferencial:

$with(DifferentialAlgebra)$:

>Definimos el anillo sobre el que actuará el algoritmo de R-G. El apartado block hace referencia al ranking empleado, en este caso hemos escogido el más óptimo computacionalmente para Maple usando doble corchete ([[...]]). Las derivadas son temporales. Hay un parámetro libre procedente del modelo de referencia (q) y otro pendiente de ajuste experimental la constante de traducción KpA:

$R := DifferentialRing(blocks = [[IA, A, B]], derivations = [t], arbitrary = [q, Kp_A])$:

>Obtenemos la base de Groebner del conjunto de polinomios P y escogemos, entre uno de los ideales que la forman, una expresión que nos sirva como futura acción de control:

$simplify(Equations(G[1], solved)[2])$

$I_A(t) = -\frac{1500qA(t)-10A(t)-1500B(t)+1680Kp_A}{150qA(t)-A(t)-150B(t)+336Kp_A}$

>Ahora comprobamos, de nuevo mediante el algoritmo de Rosenfeld-Groebner, si la acción de control elegida es capaz de reproducir el modelo de referencia:

$PC := [p1, p2, Equations(G[1], solved)[2]]$:
$GC := RosenfeldGroebner(PC, R)$:
$BelongsTo(ref, GC)$

$true$

>Al pertenecer el modelo de referencia (ref) a la base de Groebner GC, obtenida a partir del polinomio del circuito genético reducido ($p1$) y la expresión de la acción de control ($I_A(t)$), podemos concluir que el control es adecuado.

Programa MATLAB

```
%% CIRCUITO GENETICO REDUCIDO

A_0 = 1; %Cantidad inicial de proteina A
B_0 = 3; %Cantidad inicial de proteina B
T = 10; %En horas

q=0; %Modelo de referencia B=Ap + q*A

sim('genes_simplificado.slx'); %Ejecuta el SIMULINK del modelo

%% Graficas

figure
subplot(211)
plot(B.time,B.signals.values,'y','LineWidth',1.5)
grid on, hold on
plot(Ap.time,Ap.signals.values + q*A.signals.values,'k-.')
plot(A.time,A.signals.values,'LineWidth',1.5)
legend('\bf Referencia: Ap + q*B','\bf B (nM)','\bf A (nM)')
title('\fontsize{14} Comportamiento del modelo reducido')
subplot(212)

plot(IA.time,IA.signals.values,'r','LineWidth',1.5)
ylabel('\bf (nM)'),grid on
legend('\bf IA(t)')

xlabel('\bf Tiempo (Hr)')

%% SUPERFICIE DE LA ACCION DE CONTROL
%Por comodidad renombramos:
    %A(t) -> X
    %B(t) -> Y
    %IA(t) -> Z

%Valores de los parametros:
q=0;
KpA=100/60;

%Rango de valores de las proteinas A,B
x = 0:0.5:30;
y = 0:0.5:30;

%Creamos el meshgrid
[X,Y] = meshgrid(x,y);

%Escribimos la accion de control:
Z = -10.*(150.*q.*X + 168.*KpA - X - 150.*Y)./(150.*q.*X + 336.*KpA -
   X - 150.*Y);
```

```
48
49 figure
50 surf(X,Y,Z)
51 xlabel('\bf A (nM)')
52 ylabel('\bf B (nM)')
53 zlabel('\bf IA (nM)')
54 axis([0 30 0 30 -100 200])
55 title('\bf Accion de control')
```

Subsistemas en simulink

Subsistemas en esquema de Lotka Volterra

Subsistemas del esquema de simulink correspondiente a la figura (2).

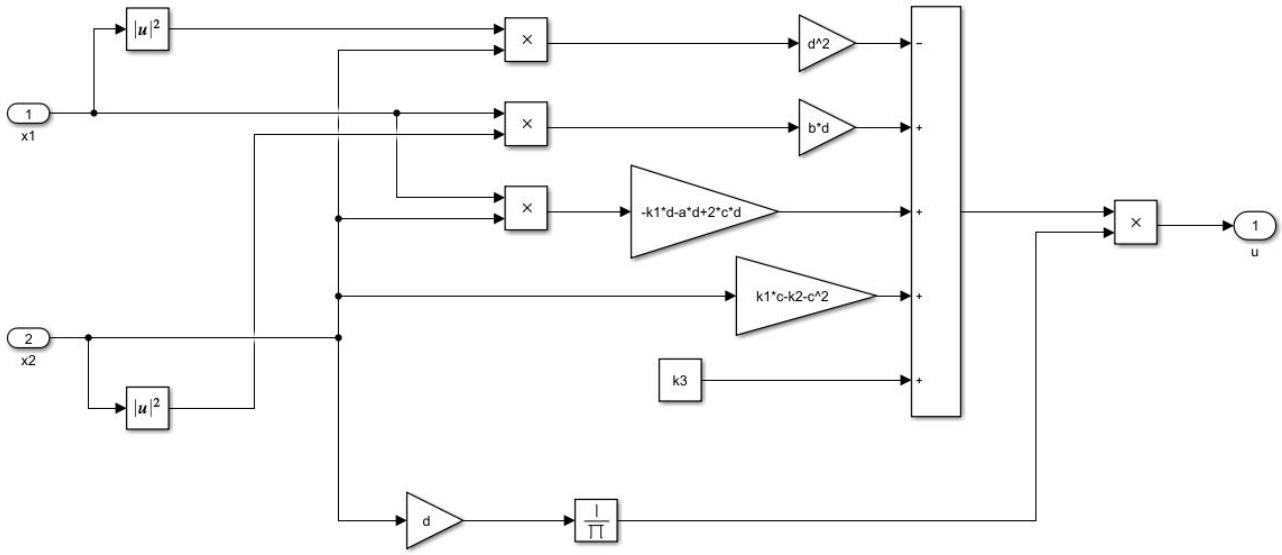


Figura 23: Subsistema de la acción de control u.

Subsistemas en esquema del circuito genético

Subsistemas del esquema de simulink correspondiente a la figura (7).

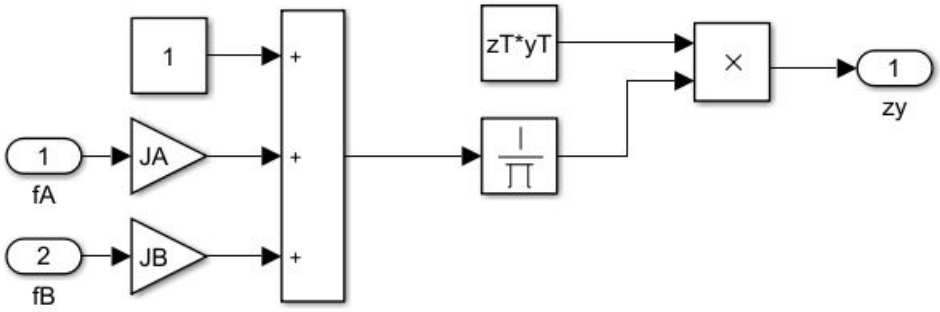


Figura 24: Subsistema de la función zy.

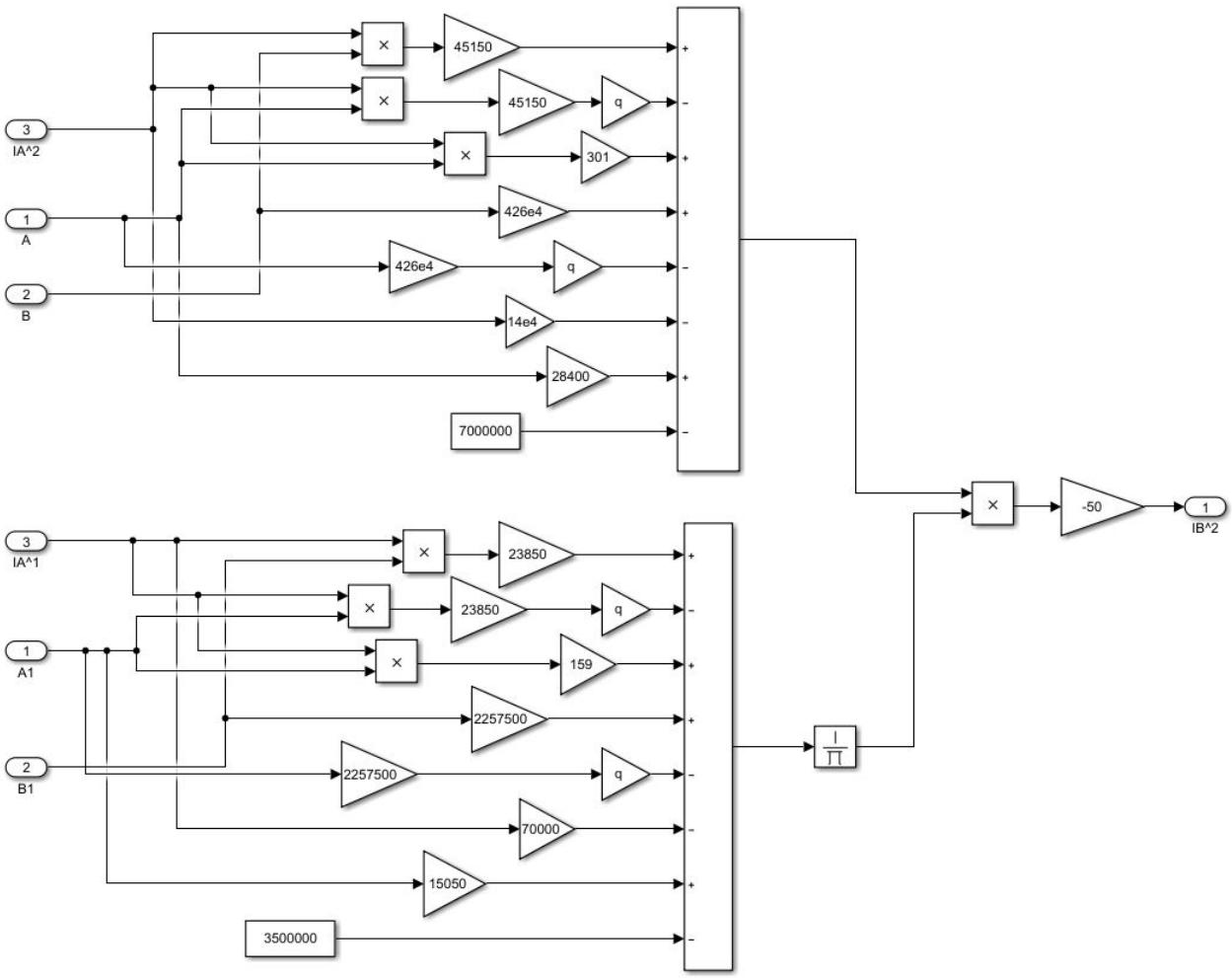


Figura 25: Subsistema de control.

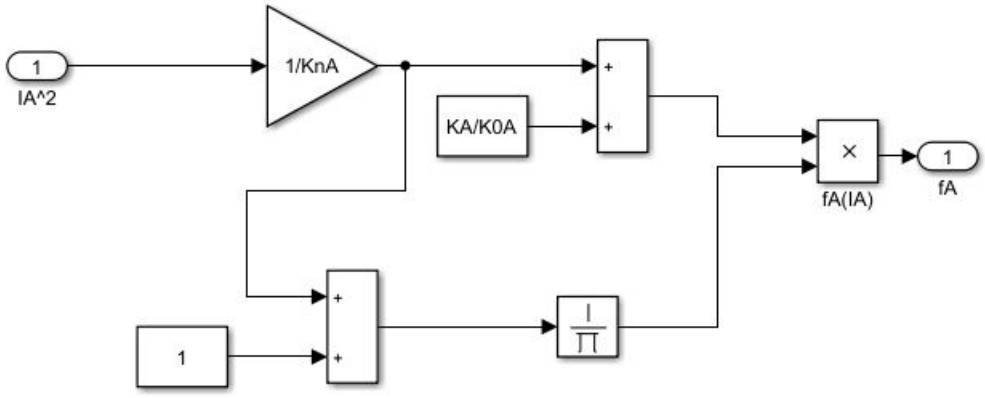


Figura 26: Subsistema de la función $f_A(I_A)$.

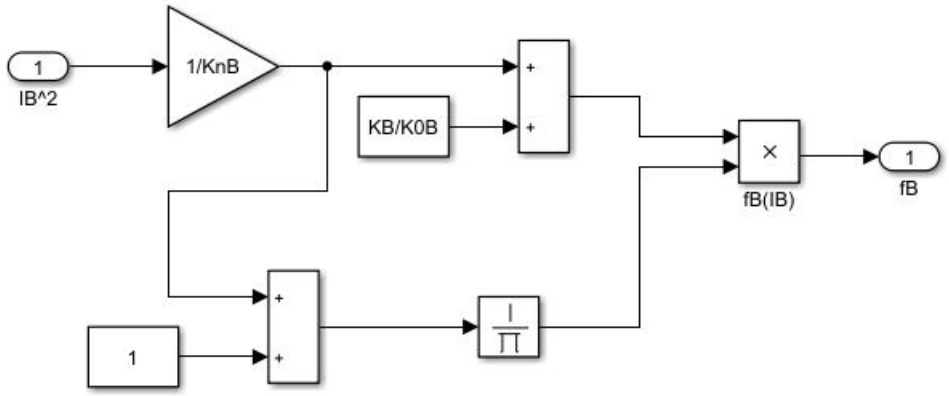


Figura 27: Subsistema de la función $f_B(I_B)$.

Subsistemas en esquema del modelo reducido

Subsistemas del esquema de simulink correspondiente a la figura (20).

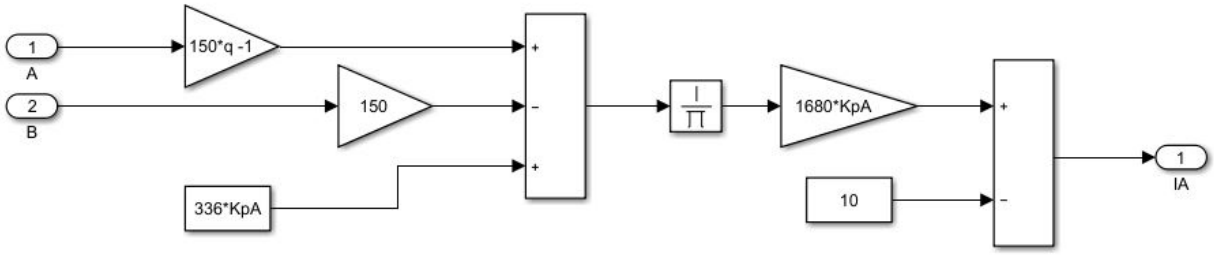


Figura 28: Subsistema de control.

Desarrollo matemático en la obtención del modelo genético

Pasos intermedios en la obtención del modelo matemático (23), (24). Modelado a partir del proceso biológico, aplicando la ley cinética de acción de masas (2) y la aproximación *Quasy steady state*, abreviada como QSSA [10].

$$\begin{aligned}\dot{C}_A^{y_0} &= K_A^{y_0}\cdot gA\cdot y - K_{-A}^{y_0}\cdot C_A^{y_0} - k_{mA}\cdot C_A^{y_0}\\ &\downarrow QSSA\\ C_A^{y_0} &= \frac{K_A^{y_0}}{K_{-A}^{y_0}+K_{mA}}\cdot y\cdot gA \triangleq \frac{1}{K_A^0}\cdot y\cdot gA\end{aligned}$$

$$\begin{aligned}\dot{C}_A^{TF} &= K_A^{+}\cdot gA\cdot I_A^{nA} - K_{nA}^{-}\cdot C_A^{TF} - K_A^{y}\cdot C_A^{TF}\cdot y + K_{-A}^{y}\cdot C_A^{y} + K_{mA}\cdot C_A^{y}\\ &\downarrow QSSA\\ C_A^{TF} &= \frac{K_{nA}^{+}}{K_{nA}^{-}}\cdot gA\cdot I_A^{nA} \triangleq \frac{1}{K_{nA}}\cdot gA\cdot I_B^{nA}\end{aligned}$$

$$\begin{aligned}\dot{C}_A^{y} &= K_A^{y}\cdot C_A^{TF}\cdot y - K_{-A}^{y}\cdot C_A^{y} - K_{mA}\cdot C_A^{y}\\ &\downarrow QSSA\\ C_A^{y} &= \frac{K_A^{y}}{K_{-A}^{y}+K_{mA}}\cdot y\cdot C_A^{TF} \triangleq \frac{1}{K_A}\cdot y\cdot C_A^{TF}\end{aligned}$$

$$gA + C_A^{y_0} + C_A^{y} + C_A^{TF} = C_A \rightarrow \quad gA = \frac{C_A}{1+\frac{y}{K_A^0}+\left(1+\frac{y}{K_A}\right)\cdot\frac{I_A^{n_A}}{K_{nA}}}$$

$$\begin{aligned}\dot{m}A &= K_{mA}\cdot C_A^{y_0} + K_{mA}\cdot C_A^{y} + K_{-A}^{z}\cdot M_A^{z} - K_A^{z}\cdot mA\cdot z + Kp_A\cdot M_A^{z} - dm\cdot mA\\ &\downarrow QSSA\\ mA &= \frac{K_{-A}^{z}+Kp_A+\omega_A}{K_A^{z}\cdot z}\cdot M_A^{z} \triangleq \frac{\nu_A}{z}M_A^{z}\end{aligned}$$

$$\begin{aligned}\dot{M}_A^{z} &= K_A^{z}\cdot mA\cdot z - K_{-A}^{z}\cdot M_A^{z} - Kp_A\cdot M_A^{z} - \omega_A\cdot M_A^{z}\\ &\downarrow QSSA\\ M_A^{z} &= \frac{K_{mA}\cdot z\cdot C_A}{dm_A\cdot\nu_A+\omega_A\cdot z}\cdot\frac{\frac{y}{K_A^0}+\frac{y}{K_A}\cdot\frac{I_A^{n_A}}{Kn_A}}{1+\frac{y}{K_A^0}+\left(1+\frac{y}{K_A}\right)\cdot\frac{I_A^{n_A}}{K_{nA}}}\end{aligned}$$

Thus:

$$\dot{A} = \frac{K_{mA}\cdot z\cdot C_A}{(dm_A\cdot\nu_A+\omega_A\cdot z)K_A}\cdot z\cdot y\cdot f_A(I_A,y) - d_A\cdot A$$

with:

$$f_A(I_A,y) = \frac{\frac{K_A}{K_A^0}+\frac{1}{K_{nA}}\cdot I_A^{n_A}}{\left(1+\frac{y}{K_A^0}\right)+\left(1+\frac{y}{K_A}\right)\frac{1}{K_{nA}}\cdot I_A^{n_A}}$$

$$\left.\begin{array}{c} z << \nu_A \\ y << K_A^0, K_A \end{array}\right\} \text{ under enough lack of resources.}$$

Then:

$$\dot{A} = \frac{K_{pA} \cdot K_{mA} \cdot c_A}{d_{mA} \cdot \nu_A \cdot K_A} \cdot z \cdot y \cdot f_A(I_A) - d_A \cdot A$$

with:

$$f_A(I_A) = \frac{\frac{K_A}{K_A^0} + \frac{1}{K_{nA}} \cdot I_A^{nA}}{1 + \frac{1}{K_{nA}} \cdot I_A^{nA}}$$

$$\begin{aligned} y_T &= y + C_A^{y_0} + C_A^y + C_B^{y_0} + C_B^y \\ z_T &= z + M_A^z + M_B^z \end{aligned}$$

$$y = \frac{y_T}{1 + \frac{C_A}{K_A} \cdot f_A(I_A) + \frac{C_B}{K_B} \cdot f_B(I_B)}$$

$$z = \frac{z_T}{1 + \frac{K_{mA} \cdot C_A}{dm_A \cdot \nu_A \cdot K_A} \cdot f_A(I_A) \cdot y + \frac{K_{mB} \cdot C_B}{dm_B \cdot \nu_B \cdot K_B} \cdot f_B(I_B) \cdot y}$$

$$z \cdot y = \frac{z_T \cdot y_T}{1 + J_A \cdot f_A(I_A) + J_B \cdot f_B(I_B)}$$

with:

$$\begin{aligned} J_A &= \frac{C_A}{K_A}\left(1 + \frac{K_{mA} \cdot y_T}{d_{mA} \cdot \nu_A}\right) \\ J_B &= \frac{C_B}{K_B}\left(1 + \frac{K_{mB} \cdot y_T}{d_{mB} \cdot \nu_B}\right) \end{aligned}$$