UNIVERSIDAD POLITÉCNICA DE VALENCIA

Departamento de Sistemas Informáticos y Computación

ALGORITMOS DE RESOLUCIÓN DE MODELOS DE ECUACIONES SIMULTÁNEAS BASADOS EN LA DESCOMPOSICIÓN QR

José Juan López Espín

Diciembre 2009

Dirigida por:
Antonio Manuel Vidal Maciá

Quisiera agradecer a mi director de tesis de master Antonio Vidal su ayuda y apoyo. Sus consejos han sido oro para mí y me han ayudado mucho a formarme como investigador. También agradecer a Pedro Alonso, administrador de la máquina Rosebud, la paciencia que ha tenido conmigo en el desarrollo de los experimentos.

También quiero dar las gracias a los profesores del master de Computación Paralela y en general del DSIC por su gran labor docente.

Gracias a todos.

Índice general

Capítulo 1

Introducción

En este trabajo se desarrollan, estudian y comparan algoritmos basados en la descomposición QR [22] para la resolución de Modelos de Ecuaciones Simultáneas (M.E.S.) mediante el estimador de Mínimos Cuadrados en Dos Etapas (MC2E). Se desarrollan y estudian versiones de los algoritmos en secuencial y en paralelo (memoria compartida) basados en la descomposición QR mediante reflexiones de Householder y rotaciones de Givens. También se desarrolla un algoritmo basado en la creación de matrices intermedias que, no correspondiendo a una ecuación del M.E.S., permite reducir el número de rotaciones de Givens necesarias en la descomposición QR de las matrices que sí están asociadas a una ecuación.

1.1. Breve introducción histórica

Los Modelos de Ecuaciones Simultáneas forman parte del conjunto de técnicas estadísticas desarrolladas a partir de los modelos de ecuaciones de regresión [24, 25] con motivo de abarcar problemas que estos, de forma aislada, no podían resolver. Los Modelos de Ecuaciones Simultáneas nacieron en la primera mitad del siglo XX [18], siendo desarrollada la mayor parte de su cuerpo teórico en los años posteriores (en unos veinte años se había desarrollado la mayor parte de lo que se conoce hoy). Su desarrollo práctico y la aplicación a problemas reales se retrasó debido al problema computacional asociado, ya que las ecuaciones podían hacerse relativamente grandes y las técnicas de resolución necesitaban de un nivel de computación mayor al prestado en su momento por los ordenadores de la época [16, 33].

Con el paso de los años el mundo de la informática iba mejorando en todos sus aspectos. Las computadoras ganaban en velocidad, memoria, prestaciones, etc., con lo que los modelos de ecuaciones simultáneas fueron tomando una gran importancia ya que su implementación experimental no se veía excesivamente mermada por la computación.

El paso gigante fue la invención del ordenador personal y la facilidad de manejo del sistema operativo, programas, compiladores, etc. Esto hizo que muchísimos económetras no se vieran en la necesidad de trabajar conjuntamente con un equipo

de especialistas informáticos, sino que pudieran desarrollar sus propios programas y manejar sus propios ordenadores.

Los modelos de ecuaciones simultáneas son métodos estadísticos que nacieron dentro del mundo de la economía [18]. De hecho tuvieron un papel muy importante en el nacimiento de la rama de econometría, puesto que daban unas expectativas enormes en la estimación de variables económicas.

Desde su nacimiento (en los años treinta) hasta su declive (a finales de los setenta, principio de los ochenta), los Modelos de Ecuaciones Simultáneas fueron tratados, estudiados y desarrollados expresamente por económetras, es decir, se desarrollaron como una rama de la economía. Por esto hasta muy recientemente todas las aplicaciones se han desarrollado en economía y todas las conclusiones se han dado en este mismo campo, hasta el punto de entenderse esta herramienta como una herramienta económica y no estadística.

En los años cincuenta y sesenta, el peso de la econometría y de los modelos de ecuaciones simultáneas era inmenso. Se pensaba que se podría modelar cualquier proceso econométrico por complejo que este pareciera. Se desarrollaron modelos de ecuaciones simultáneas gigantescos que pretendían modelar la economía de un producto, de una región, de un país... Solo le ponía fronteras a dichos modelos la capacidad computacional del momento.

A principios de los setenta había modelos a escala mundial y las computadoras ya daban respuesta a estos modelos.

Pero la crisis del petróleo de mediados de los setenta dio un vuelco total a la econometría. Ningún modelo económico fue capaz de predecir tal desastre y, lo que es aún peor, a partir de tal punto muchos de ellos se resintieron (en el sentido de que no se ajustaban bien a la nueva situación económica) y hubo que reajustarlos, cambiarlos o, simplemente, desecharlos.

En este punto se introdujo un nuevo pensamiento en la economía y más concretamente en el mundo de la econometría. Esta nueva corriente pensaba que no era posible modelar a gran escala debido a que el mundo de la economía es un mundo caótico (en el que un atentado en un cierto instante puede cambiar el curso de la economía mundial) y el problema era que las ecuaciones simultáneas, que surgieron por tener una precisión que la regresión no daba, eran muy sensibles a elementos que por naturaleza son caóticos.

A finales de los setenta y principios de los ochenta se comenzaron a desechar modelos de ecuaciones simultáneas grandes (y han seguido en desuso hasta no hace mucho) debido a que al utilizar muchas variables en los modelos, se tenía más posibilidades de que alguna de ellas sufriera alteraciones y contaminara al modelo entero. La filosofía era usar pocas ecuaciones y pocas variables pero que fueran las verdaderamente importantes para la predicción. Con esto no se aseguraba que las estimaciones fueran precisas, pero sí que no estaban influenciadas por las fluctuaciones de otras variables de menos peso en el modelo.

Y como hasta el momento los modelos de ecuaciones simultáneas solo se habían usado en economía, cayeron en desuso. Se usaban poco y de tamaño pequeño (con

lo cual el nivel de computación exigido también era pequeño). Además, como se asumía el error que se cometía, no era necesario el usar técnicas desarrolladas para los modelos de ecuaciones simultáneas sino que simplemente se resolvían aplicando técnicas de regresión (Mínimos Cuadrados Ordinarios) aceptando que se estaba cometiendo un error de partida y que dichas técnicas vienen acompañadas de un sesgo importante. Pero merecía la pena puesto que el error no era significativo frente al ya asumido y porque el problema se simplificaba.

Actualmente se abre una nueva esperanza para los modelos de ecuaciones simultáneas. Otras ramas de tipo científico han comenzado a usar estas técnicas estadísticas para dar respuesta a problemas que no pueden ser resueltos con modelos de regresión. Por ejemplo, en el estudio de redes y el movimiento de paquetes a través de nodos [26], se ha comprobado que los modelos de ecuaciones simultáneas dan buena respuesta a la hora de modelar dicho problema, con lo que se pueden predecir tiempos de espera, tamaño de colas en los distintos nodos, etc. También se han desarrollado trabajos en medicina [23, 30, 31], en psicología [27], e incluso para modelar el tráfico aéreo en Nueva York [32] o el número de divorcios con respecto a los ingresos femeninos [36].

Básicamente un Modelo de Ecuaciones Simultáneas es un conjunto de ecuaciones de regresión donde existe influencia simultánea entre variables y ecuaciones, y donde una variable que está en una ecuación (de dependiente o de explicativa) puede aparecer en otras ecuaciones [25]. Se supone que una variable solo aparecerá como dependiente en una ecuación, aunque puede aparecer como explicativa en varias de ellas.

1.2. Estado del arte

Con lo que respecta a sofware desarrollado para la resolución de Modelos de Ecuaciones Simultáneas, podemos hablar de trabajos anteriores refiriéndonos a software de tipo comercial (o en algunos casos libre) en el cual se implementen las herramientas de resolución de Modelos de Ecuaciones Simultáneas.

En marzo de 1994, QMS inició una revolución en el software de econometría con el lanzamiento de Eviews 1.0 [9], con una moderna interface gráfica orientada a objetos para el usuario. Actualmente está implementada su versión 7.0.

Eviews lleva incorporadas la gran mayoría de técnicas de resolución de regresión y de modelos de ecuaciones simultáneas (además de series temporales y demás partes de la econometría). Sin embargo, no contempla herramientas de búsqueda del mejor modelo. Tampoco hasta ahora ha desarrollado una versión paralela de su software.

Otro programa destinado también a la econometría y de difusión gratuita es Ox [4] (OxEdit, OxMetric, y demás productos) perteneciente a doornik y que se puede descargar directamente de su web `www.doornik.com/download.html`. En este caso se disponen de menos herramientas que en el anterior (con las mismas carencias).

Actualmente muchos paquetes estadísticos llevan implementado Mínimos Cua-

drados en dos Etapas (MC2E), que es uno de los estimadores más usados en los modelos de ecuaciones simultáneas y que será el utilizado en este trabajo. Sin embargo su implementación no es para resolución de estos sistemas sino para resolver una ecuación de regresión usando la información almacenada en otras variables que no entren en el modelo. Es decir, exactamente lo que hace MC2E al resolver una ecuación en un modelo pero de forma aislada. Por ejemplo la versión de SPSS 15.0 [10] ya lleva implementado MC2E.

En [29] se desarrollan y estudian algoritmos para estimadores de información completa de los M.E.S., más concretamente MC3E [24, 25]. Sin embargo, hasta donde nosotros sabemos, no han sido abordados por otros autores desde el prisma computacional algoritmos de información limitada (que resuelvan cada una de las ecuaciones por separado), por lo que este se convierte en el objetivo del presente trabajo.

1.3. Objetivos del presente trabajo

Los **objetivos** del presente trabajo son:

- **Resolver Modelos de Ecuaciones Simultáneas** de manera más eficiente reduciendo el tiempo de ejecución y los recursos en memoria utilizando reflexiones de Householder y rotaciones de Givens.

- **Comparar los algoritmos desarrollados** de manera que se obtenga un criterio en la elección de uno u otro método en la resolución de M.E.S.

- **Mejorar dichos algoritmos** con la introducción de matrices que no correspondiendo a ecuaciones del problema, reducen el número de operaciones necesarias en la descomposición QR de las matrices que sí corresponden a las ecuaciones del problema.

- **Paralelizar** los algoritmos descritos en memoria compartida y estudiar su speed-up.

Esto permitirá abordar Modelos de Ecuaciones Simultáneas con gran número de ecuaciones y de variables. La gran cantidad de tiempo de ejecución requerido por este tipo de modelos y el auge actual de los multicores, hace importante desarrollar versiones para memoria compartida.

Además de desarrollar algoritmos que se ejecuten en multicore, la finalidad de la obtención de dichos algoritmos es que sean lo más portables posible y que puedan ser ejecutados en diversos sistemas. Para ello se utilizarán llamadas a las librerías de LAPACK [5, 12] y BLAS [1, 17].

1.4. Aportaciones

Las aportaciones hechas por este trabajo y que tienen un carácter novedoso se pueden clasificar en los siguientes puntos:

- Una primera aportación ha sido el estudio y comparación teórica y experimental y desarrollo en paralelo de algoritmos para la resolución de M.E.S. basados en estimadores de información limitada. De hecho, solo tenemos constancia de que se hayan estudiado y desarrollado algoritmos paralelos por otros autores para estimadores de información completa [29].

- La segunda aportación ha sido utilizar la descomposición QR para acelerar los algoritmos descritos, estudiándolos y comparándolos a continuación teórica y experimentalmente. Se han utilizado para ello técnicas e ideas usadas en modelos de regresión, pero que hasta ahora no se habían usado en M.E.S. Las técnicas usadas son las reflexiones de Householder, las rotaciones de Givens [22] y creación de árboles para reducir el número de retriangularizaciones [37].

1.5. Metodología

La metodología utilizada en el presente trabajo ha sido la siguiente:

- Se han desarrollado algoritmos para la resolución de M.E.S. realizando un estudio teórico del tiempo de ejecución. El estudio del tiempo de ejecución permite la comparación del coste computacional de los algoritmos.

- Se han desarrollado algoritmos paralelos de los algoritmos ya estudiados, puesto que el coste computacional es alto.

- En todos los casos, se han implementado dichos algoritmos realizando a continuación un estudio experimental.

- Se ha contrastado el estudio experimental con el estudio teórico.

En los diferentes estudios teóricos, se han considerado aproximaciones para facilitar su comparación. En el estudio experimental se han medido los tiempos variando diferentes parámetros (como el tamaño de la muestra, el número de ecuaciones ...)

1.6. Contenido de la tesis de master

El presente trabajo se estructura en 6 capítulos. El primero, como ya se ha visto, introduce al lector en el problema a abordar.

El capítulo 2 describe las herramientas básicas usadas en el trabajo. Se describe el cluster utilizado para los experimentos y las librerías software usadas en ellos.

También se describen herramientas matemáticas que serán utilizadas a lo largo del trabajo.

El capítulo 3 introduce los Modelos de Ecuaciones Simultáneas y el estimador para el cual se desarrollarán los algoritmos que se abordarán en éste trabajo, Mínimos Cuadrados en Dos Etapas (MC2E).

El capítulo 4 analiza diferentes versiones del algoritmo basado en el estimador MC2E, utilizando la descomposición QR. Se introduce la idea de obtener la descomposición QR de la matriz asociada a una ecuación a partir de la descomposición QR usada en los cálculos hechos al principio del algoritmo, evitando realizar una descomposición QR completa en cada ecuación y obteniendo por tanto el consiguiente ahorro computacional. También se desarrollan versiones paralelas en memoria compartida de dichos algoritmos.

En el capítulo 5 se estudian mejoras de los algoritmos planteados en el capítulo 4 mediante el diseño de un árbol donde hay nodos que corresponden a ecuaciones del Modelo de Ecuaciones Simultáneas y nodos ficticios que se introducen para reducir el número de operaciones a realizar al obtener la descomposición QR. Se estudia experimentalmente para qué valores del tamaño del problema es rentable la introducción de nodos ficticios y cual es la mejora obtenida. También se estudia la paralelización en memoria compartida.

El capítulo 6 recoge las principales conclusiones obtenidas con relación a esta tesis de máster y se relacionan algunas posibles líneas futuras de trabajo.

Capítulo 2

Herramientas básicas

Se describen en este capítulo las diferentes herramientas computacionales, tanto software como hardware, utilizadas en el trabajo. También se hace una descripción de varias herramientas matemáticas básicas que serán utilizadas como base para diferentes algoritmos en el presente trabajo. Éste es el caso de Mínimos Cuadrados y la descomposición QR.

2.1. Herramientas computacionales

Se resume en esta sección las máquinas utilizadas para el desarrollo y estudio de los algoritmos propuestos, y el software utilizado en dichas máquinas para este fin.

2.1.1. Hardware

En este trabajo se ha utilizado el cluster Rosebud perteneciente al grupo INCO2 del departamento de Sistemas Informáticos y Computación de la Universidad Politécnica de Valencia. Rosebud es un cluster heterogéneo compuesto por 8 ordenadores conectados mediante una red Fast-Ethernet. Los 8 ordenadores se agrupan en 4 pares de computadores, diferentes entre sí en cuanto a potencia de cálculo y en cuanto a arquitectura. Los dos primeros, denominados rosebud01 y rosebud02, están formados por procesadores Pentium IV con velocidades de 1.6 GHz y 1.7 GHz respectivamente, y una memoria RAM de 1 Gbyte. Los computadores del segundo par, denominados rosebud03 y rosebud04, son biprocesadores Xeon con una velocidad de 2.2 GHz y con 3.5 Gbyte de memoria RAM (figura 2.1 izquierda). Estos dos primeros pares presentan una arquitectura i386. El tercer par está formado por dos máquinas Fujitsu Primergy RXI600 (denotados por rosebud05 y rosebud06), cada una con 4 procesadores Dual-Core Intel Itanium2 a 1.4 GHz compartiendo 8 GByte de memoria RAM (figura 2.1 derecha). El procesador Itanium2 presenta una arquitectura de 64 bits. El cuarto par está formado por Intel Core 2 Quad, presentando cada core una velocidad de 2.66 GHz y compartiendo 4 GByte de memoria RAM. En total, en el cluster Rosebud se cuenta con 30 núcleos computacionales.

En este trabajo, todos los experimentos han sido realizados en rosebud05 y rosebud06, por lo que se omitirá el número nombrando tan solo la máquina Rosebud.

Figura 2.1: Máquina Rosebud. A la izquierda los nodos rosebud03 y rosebud04, a la derecha el nodo rosebud05.

2.1.2. Software

Las herramientas software usadas para el presente trabajo se pueden dividir en dos grupos diferentes, uno, los lenguajes de programación utilizados, y otro, las librerías usadas.

Como lenguajes de programación, se ha utilizado ANSI C para la programación secuencial y el API OpenMP para memoria compartida [11, 13, 14, 35].

OpenMP

La librería OpenMP es un API que permite añadir concurrencia a los procesos mediante paralelismo de memoria compartida. Con OpenMP es posible desarrollar programas a un nivel superior que hacerlo creando hilos controlando la concurrencia de éstos. Con las directivas de OpenMP se hacen transparente las operaciones de concurrencia de threads así como la creación, inicialización, destrucción, etc., de threads.

OpenMP consta de tres componentes API principales: directivas del compilador, rutinas de librería de tiempo de ejecución y variables de entorno. Es una librería portable, pues está especificada para los lenguajes C/C++ y Fortran y se ha implementado en múltiples plataformas incluyendo Unix y Windows.

Librerías de álgebra lineal

Debido a la gran importancia que tiene el álgebra lineal dentro de los problemas que se puedan plantear en cualquiera de las ciencias, ingenierías, e incluso en

aplicaciones de índole comercial, a principios de los años setenta se comenzaron a desarrollar aplicaciones para resolver problemas de este tipo, de manera más eficiente tanto en secuencial como en paralelo.

La librería BLAS [1, 17] se compone de funciones que se basan en la construcción de bloques para efectuar operaciones básicas entre vectores y matrices. BLAS está construido en tres niveles: el nivel 1, que efectúa operaciones vector-vector y es llamado BLAS 1, el nivel 2, que efectúa operaciones matriz-vector, que es llamado BLAS 2, y el nivel 3, que efectúa operaciones matriz-matriz y es llamado BLAS 3. Esta herramienta es eficiente, portable y de amplia disponibilidad, por lo que se utiliza comúnmente en el desarrollo de software del álgebra lineal de altas prestaciones. La librería PBLAS [7, 15] contiene versiones en paralelo de las funciones desarrolladas en BLAS.

La librería LAPACK [5, 12] proporciona rutinas para resolver sistemas de ecuaciones, problemas de mínimos cuadrados, problemas de valores propios, vectores propios y valores singulares. También proporciona rutinas para factorización de matrices, tales como LU, Cholesky, QR, etc., tanto para matrices con estructuras específicas o matrices generales.

2.2. Herramientas matemáticas

Se explican en este apartado varias herramientas matemáticas básicas que serán utilizadas a lo largo del presente trabajo. En primer lugar se define la expresión del estimador de Mínimos Cuadrados añadiéndole de aquí en adelante, para evitar confusión con otros tipos de Mínimos Cuadrados que se estudiarán en este trabajo, el apelativo de Ordinarios. A continuación se trata la descomposición QR y dos métodos para su obtención, reflexiones de Householder y rotaciones de Givens.

2.2.1. Mínimos Cuadrados Ordinarios (MCO)

La expresión del estimador de Mínimos Cuadrados para el problema de regresión múltiple es descrita a continuación. Se considera una única ecuación donde la variable expresada o variable dependiente será denotada por z, y las variables en la expresión como explicativas o independientes, serán denotadas por $w_1, \ldots, w_n$. La variable aleatoria de ruido blanco será denotada por v. Así, tenemos:

$$z_t = \beta_1 w_{1,t} + \ldots + \beta_n w_{n,t} + v_t \tag{2.1}$$

siendo $z_t, w_{1,t}, \ldots, w_{n,t}, v_t$ elementos de las variables $z, w_1, \ldots, w_n, v$ respectivamente, con $t = 1, ..., d$ y siendo d el tamaño muestral.

La ecuación 2.1 puede expresarse en forma matricial de la siguiente manera:

$$z = W\beta + v \tag{2.2}$$

siendo:

$$z = \begin{pmatrix} z_1 \\ \vdots \\ z_d \end{pmatrix}, \; W = \begin{pmatrix} w_{1,1} & \dots & w_{n,1} \\ \vdots & & \vdots \\ w_{1,d} & \dots & w_{n,d} \end{pmatrix}, \; \beta = \begin{pmatrix} \beta_1 \\ \vdots \\ \beta_n \end{pmatrix}, \; v = \begin{pmatrix} v_1 \\ \vdots \\ v_d \end{pmatrix} \tag{2.3}$$

La expresión del estimador MCO es $(W^TW)^{-1}W^Tz$. El vector resultante es el estimador de β, que denotaremos por $\hat{\beta}$. Con esta estimación se consigue calcular los coeficientes que relacionan la variable dependiente con las explicativas. Una vez obtenidos los coeficientes se pueden hacer estimaciones de la variable z a partir de W de la forma $\hat{z} = W\hat{\beta}$.

2.2.2. Descomposición QR

La descomposición QR de una matriz es una descomposición matricial que expresa la matriz original como producto de dos matrices tal como expresa la siguiente proposición [22].

Proposición Dada una matriz $A \in \mathbb{R}^{m\times n}, m \geq n$, $Rango(A) = n$, existen matrices $Q \in \mathbb{R}^{m\times m}$ ortogonal y $R \in \mathbb{R}^{m\times n}$ triangular superior de la forma $\begin{pmatrix} R_1 \\ 0 \end{pmatrix} \begin{matrix} n \times n \\ (m-n) \times n \end{matrix}$, donde R_1 es una matriz triangular superior, tales que $A = QR$.□

Por lo tanto, se necesitan algoritmos para calcular la matriz Q y la matriz R a partir de una matriz dada A de forma que ambas matrices cumplan las condiciones de la proposición. A continuación se exponen dos métodos para el cálculo de la descomposición QR de una matriz. Cada uno de ellos tiene características diferentes y serán utilizados en diferentes algoritmos para la resolución de M.E.S.

Descomposición QR mediante reflexiones de Householder

Una transformación de Householder o reflexión de Householder es una transformación que refleja el espacio con respecto a un hiperplano determinado. Esta propiedad se puede utilizar para realizar la transformación QR de una matriz si tenemos en cuenta que es posible diseñar la matriz de Householder de manera que un vector elegido quede con una única componente no nula tras ser transformado (es decir, premultiplicando por la matriz de Householder). Gráficamente, esto significa que es posible reflejar el vector elegido respecto de un hiperplano de forma que el reflejo quede sobre uno de los ejes de la base cartesiana.

La elección del hiperplano de reflexión y la construcción de la matiz de Householder asociada, es de la forma siguiente: Sea $x \in \mathbb{R}^m$ un vector columna arbitrario tal que $||x||_2 = |\alpha|$, donde α es un escalar. Entonces, siendo e_1 el vector $(1, 0, ..., 0)^T$, se define $u = x \pm \alpha e_1, v = \frac{u}{||u||_2}$ y $Q = I - 2vv^T$, donde v es un vector unitario perpendicular al hiperplano de reflexión elegido, Q es una matriz de Householder asociada a dicho hiperplano y $Qx = (\alpha, 0, \cdots, 0)^T$. En [22] puede verse una descripción del proceso en detalle incluyendo la elección óptima del signo.

Esta propiedad se puede usar para transformar gradualmente los vectores columna de una matriz A en una matriz triangular superior. En primer lugar se multiplica A con la matriz de Householder Q_1 que obtenemos al elegir como vector x la primera columna de la matriz. Esto proporciona una matriz $Q^T A$ con ceros en la primera columna (excepto el elemento de la primera fila):

$$Q_1 A = \begin{pmatrix} \alpha_1 & \star & \dots & \star \\ 0 & & & \\ \vdots & & A' & \\ 0 & & & \end{pmatrix}$$

El procedimiento se puede repetir para A' (que se obtiene de A eliminando la primera fila y columna), obteniendo así una matriz de Householder Q'_2. Hay que tener en cuenta que Q'_2 es de menor tamaño ($(m-1) \times (m-1)$) que Q_1. Para conseguir que esta matriz opere con $Q_1 A$ en lugar de A' es necesario expandirla hacia arriba y hacia la izquierda, completando con un uno en la diagonal, o en general:

$$Q_k = \begin{pmatrix} I_{k-1} & 0 \\ 0 & Q'_k \end{pmatrix}$$

donde I_k es la matriz identidad de dimensión $k \times k$.

Tras repetir el proceso t veces, donde $t = \text{mín}(m-1, n), R = Q_t \cdots Q_2 Q_1 A$ es una matriz triangular superior. Tomando $Q = Q_1 Q_2 \cdots Q_t, A = QR$ es una descomposición QR de la matriz A.

El coste de calcular la descomposición QR mediante el método de Householder de una matriz $A \in \mathbb{R}^{m \times n}, m \geq n$, viene dado por la siguiente expresión [6, 22]:

$$C_{HH}(n, m) = \frac{2}{3} n^2 (3m - n) \tag{2.4}$$

donde se ha contado el número de flops (operaciones en coma flotante) necesarios para completar la descomposición QR de la matriz A.

Descomposición QR mediante rotaciones de Givens

Una rotación de Givens situada en el plano (i, j) que reduce a cero el elemento $b_{j,k}$ cuando se aplica por la izquierda a la matriz $B \in \mathbb{R}^{m \times n}$ será denotada por $G_{i,j}^{(k)}$ donde $1 \leq i, j \leq m$ y $1 \leq k \leq n$. La rotación $G_{i,j}^{(k)} B$ afecta únicamente a la i-ésima y j-ésima filas de B. Los cambios en dichas filas pueden ser expresados como sigue:

$$\begin{pmatrix} c & s \\ -s & c \end{pmatrix} \begin{pmatrix} b_{i,:} \\ b_{j,:} \end{pmatrix} = \begin{pmatrix} \tilde{b}_{i,1} & \dots & \tilde{b}_{i,k} & \dots & \tilde{b}_{i,n} \\ \tilde{b}_{j,1} & \dots & \tilde{b}_{j,k} & \dots & \tilde{b}_{j,n} \end{pmatrix} \tag{2.5}$$

donde $b_{j,k} \neq 0$, $c^2 + s^2 = 1$, $c = b_{i,k}/\tau$, $s = b_{j,k}/\tau$, $\tau^2 = b_{i,k}^2 + b_{j,k}^2$, $\tilde{b}_{i,k} = \tau$ y $\tilde{b}_{j,k} = 0$.

Así, la matriz $G_{i,j}^{(k)} \in \mathbb{R}^{m \times m}$ queda definida de la siguiente forma:

$$G_{i,j}^{(k)} = \begin{array}{c} \\ i \rightarrow \\ \\ j \rightarrow \\ \\ \end{array} \begin{array}{c} \begin{array}{ccccccc} & & i & & j & & \\ & & \downarrow & & \downarrow & & \end{array} \\ \begin{pmatrix} 1 & \dots & 0 & \dots & 0 & \dots & 0 \\ 0 & \dots & c & \dots & s & \dots & 0 \\ 0 & & \vdots & \ddots & \vdots & & 0 \\ 0 & \dots & -s & \dots & c & \dots & 0 \\ 0 & \dots & 0 & \dots & 0 & \dots & 1 \end{pmatrix} \end{array} \tag{2.6}$$

Puesto que esta matriz es ortogonal, si multiplicamos B por el conjunto de rotaciones de Givens que reducen a ceros los elementos por debajo de la diagonal principal en el orden descrito en la figura 2.2, habremos obtenido la descomposición QR de B. En la figura 2.2 se muestra el orden de eliminación de elementos en el proceso de descomposición QR mediante rotaciones de Givens de una matriz $A \in \mathbb{R}^{10\times 5}$. La entrada i (i=1,...,35) indica el elemento reducido a cero en la i-ésima rotación.

$$\begin{pmatrix} \bullet & \bullet & \bullet & \bullet & \bullet \\ 9 & \bullet & \bullet & \bullet & \bullet \\ 8 & 17 & \bullet & \bullet & \bullet \\ 7 & 16 & 24 & \bullet & \bullet \\ 6 & 15 & 23 & 30 & \bullet \\ 5 & 14 & 22 & 29 & 35 \\ 4 & 13 & 21 & 28 & 34 \\ 3 & 12 & 20 & 27 & 33 \\ 2 & 11 & 19 & 26 & 32 \\ 1 & 10 & 18 & 25 & 31 \end{pmatrix}$$

Figura 2.2: Orden de anulación en la descomposición QR aplicada a $A \in \mathbb{R}^{10\times 5}$.

El número de rotaciones de Givens necesarias para calcular la descomposición QR de una matriz $A \in \mathbb{R}^{m\times n}$ es $\sum_{i=1}^{n}(m-i) = \frac{n}{2}(2m-n-1)$, y Q^T es definida como sigue:

$$Q^T = \prod_{i=1}^{n}\prod_{j=1}^{m-i} G_{m-j,m-j+1}^{(i)} \tag{2.7}$$

La construcción de una rotación de Givens requiere seis flops. El mismo tiempo es el empleado en aplicar la rotación a dos elementos, por lo que si tenemos en cuenta que al aplicar Givens en la expresión 2.5 no se calcula la primera columna (a $b_{i,k}$ se le asigna el valor de τ y $b_{j,k}$ se hace cero), el número de operaciones total será $6(n-k)$.

La complejidad de calcular 2.7 viene dada en la siguiente expresión:

$$C_G(n,m) = 6\sum_{i=1}^{n}(m-i)(n-i+1) = n(3m(n+1)-n^2-3n-2) \approx n^2(3m-n) \quad (2.8)$$

En [22] puede verse una descripción más detallada del algoritmo de la descomposición QR utilizando rotaciones de Givens.

2.3. Conclusiones

En este capítulo se han descrito las herramientas básicas que serán usadas a lo largo del presente trabajo. Se ha descrito el hardware donde se han realizado los experimentos y las librerías software que han sido utilizadas.

También se describen, como herramientas matemáticas que serán utilizadas a lo largo del trabajo, el método de Mínimos Cuadrados Ordinarios, la descomposición QR de una matriz y los algoritmos para obtenerla (transformación de Householder y rotaciones de Givens).

Capítulo 3

Descripción teórica

3.1. Descripción teórica

En esta sección se da una descripción básica de los Modelos de Ecuaciones Simultáneas, los tipos de variables que aparecen en ellos y el estimador usado en este trabajo para resolverlos.

3.1.1. Modelo de Ecuaciones Simultáneas

Tres tipos de variables aparecen en un sistema de ecuaciones simultáneas representadas por Y, X y u (se definen al final de la sección). Los parámetros que se usarán en las descripción teórica dada a continuación y en los algoritmos propuestos a lo largo del trabajo, serán los siguientes:

- El tamaño de la muestra, d.
- El número de variables endógenas, N.
- El número de variables exógenas, K.
- El número de variables endógenas en la ecuación i, n_i (o lo que es lo mismo, el número de coeficientes distintos de cero en la fila i-ésima de la matriz B).
- El número de variables exógenas en la ecuación i, k_i (o lo que es lo mismo, el número de coeficientes distintos de cero en la fila i-ésima de la matriz Γ).
- La matriz de variables exógenas X, de dimensión $d \times K$.
- La matriz de variables endógenas Y, de dimensión $d \times N$.

El esquema de un modelo con N ecuaciones, N variables endógenas y K variables exógenas en forma matricial es:

$$BY^T + \Gamma X^T + u^T = 0 \tag{3.1}$$

siendo:

$$Y = (y_1 \dots y_N) \ , \ X = (x_1 \dots x_K) \ , \ u = (u_1 \dots u_N)$$
$$B = \begin{pmatrix} \beta_{1,1} & \dots & \beta_{1,N} \\ & \dots & \\ \beta_{N,1} & \dots & \beta_{N,N} \end{pmatrix}, \ \Gamma = \begin{pmatrix} \gamma_{1,1} & \dots & \gamma_{1,K} \\ & \dots & \\ \gamma_{N,1} & \dots & \gamma_{N,K} \end{pmatrix} \tag{3.2}$$

matrices densas con valores reales, siendo $\beta_{i,i} = -1 \ \forall \ i = 1, ..., N$.

La ecuación 3.2 se dice que está en forma estructural y puede ser expresada también en ecuaciones:

$$\begin{aligned} y_1 &= \gamma_{1,1}x_1 + \ldots + \gamma_{1,K}x_K + \beta_{1,2}y_2 + \beta_{1,3}y_3 + \ldots + \beta_{1,N}y_N + u_1 \\ y_2 &= \gamma_{2,1}x_1 + \ldots + \gamma_{2,K}x_K + \beta_{2,1}y_1 + \beta_{2,3}y_3 + \ldots + \beta_{2,N}y_N + u_2 \\ &\dots \\ y_N &= \gamma_{N,1}x_1 + \ldots + \gamma_{N,K}x_K + \beta_{N,1}y_1 + \ldots + \beta_{N,N-1}y_{N-1} + u_N \end{aligned} \tag{3.3}$$

donde $x_1, x_2, \dots, x_K$ son variables exógenas, $y_1, y_2, \dots, y_N$ son variables endógenas, y $u_1, u_2, \dots, u_N$ son variables de ruido blanco. Todas ellas son vectores de dimensión d, siendo d el tamaño de la muestra.

El **objetivo** al resolver un M.E.S. es **estimar** $\gamma_{1,1}, \dots, \gamma_{N,K}, \beta_{1,2}, \beta_{1,3}, \dots, \beta_{N,N-1}$ a partir de los datos tomados en cada una de las variables. Es decir, se dispone de d datos de las K variables exógenas $x_1, x_2, \dots, x_K$ y d datos de las N variables endógenas $y_1, y_2, \dots, y_N$ de un M.E.S., y lo que se pretende es estimar a partir de dichos datos, los coeficientes del modelo que las relaciona. Dichos coeficientes son los dados en las matrices B y Γ. Por lo tanto, el **objetivo** de la tesis de master es **estimar** los coeficientes dados en las matrices B y Γ a partir de los datos de las variables dados en las matrices X e Y, **reduciendo el tiempo de ejecución mediante el uso de la descomposición QR**.

Cada ecuación tiene una variable endógena que está despejada en la parte izquierda y que llamaremos endógena principal. Cada ecuación está diseñada para expresar su endógena principal respecto a un conjunto de variables endógenas y exógenas.

Si se despejan las endógenas principales, es decir, si se descompone B de la forma $B = diag(-1, ..., -1) + \tilde{B}$, la expresión 3.2 queda de la siguiente forma:

$$Y^T = \tilde{B}Y^T + \Gamma X^T + u^T \tag{3.4}$$

siendo todas las matrices las de antes y teniendo la matriz $\tilde{B}$ ceros en la diagonal principal.

El modelo estructural (ecuacion 3.1) se puede expresar también en forma reducida como sigue:

$$Y = X\Pi + v \tag{3.5}$$

con:

$$\Pi^T = -B^{-1}\Gamma \ , \ v^T = -B^{-1}u^T \tag{3.6}$$

Las variables que aparecen en un Modelo de Ecuaciones Simultáneas son las siguientes:

- Endógenas: Son variables que influyen en el modelo y se ven influenciadas por él. Las representaremos por y y supondremos que el total de variables, N, coincide con el total de ecuaciones (a este tipo de sistemas se les denomina completos).

- Exógenas: Son variables que influyen en el modelo pero no se ven influenciadas por él. Las representaremos con x.

- Predeterminadas: Son aquellas formadas por las exógenas y endógenas retardadas. Una variable endógena retardada es una variable endógena que entra en una ecuación con datos retardados en el tiempo. Por ejemplo, en un sistema de predicción de la oferta y demanda de un producto [34], el precio del mes pasado puede entrar en la ecuación de predicción de la demanda, y al ser un valor pasado no puede verse influenciado y variar por culpa del sistema, pero tampoco es una variable exógena, puesto que el precio actual sí se ve influenciado por el sistema. Es, por tanto, una variable predeterminada. Es decir, las variables predeterminadas son una extensión de las exógenas. Por comodidad de notación asumiremos que todas las variables predeterminadas en el sistema son exógenas.

- Ruido blanco: Son las variables de error que se acepta que existen en un modelo de ecuaciones simultáneas. Una variable de ruido blanco es una variable cuyos datos están idénticamente distribuidos según una normal de media cero y de desviación típica constante. Obviamente, cuanto menor sea dicha desviación típica menores serán los errores en la relación entre las variables endógenas y sus ecuaciones en el modelo y, por lo tanto, mejor será su predicción, y más fiable su uso y las conclusiones derivadas a partir de él.

- Dependiente: Es la variable que es expresada en función de otras variables en una ecuación de regresión. En los M.E.S. se utiliza la notación de variable dependiente o endógena principal para hacer referencia, en una ecuación, a la variable que está despejada en función de otras variables.

- Explicativas: Son las variables que se sitúan en la expresión de una variable dependiente en una ecuación de regresión. En los M.E.S. se utiliza dicha notación para hacer referencia a las variables tanto endógenas como exógenas que entran en una ecuación y que no son la endógena principal.

Para aplicar MCO sin obtener estimadores sesgados es necesario que no haya correlación entre el término aleatorio o variable de ruido blanco y las variables

explicativas de la ecuación. En la teoría básica de los M.E.S. se ha demostrado que si una variable endógena entra de explicativa en una ecuación cuya principal es explicativa en la ecuación donde esta endógena es endógena principal, entonces existe correlación entre el término aleatorio y ella misma, y por lo tanto no se puede aplicar MCO. Esta es la razón por la que en los M.E.S. no se aplica directamente MCO y hay que buscar otros estimadores.

3.1.2. El problema de la identificación

Antes de resolver una ecuación en un modelo de ecuaciones simultáneas hay que tener en cuenta que no todas las ecuaciones pueden ser resueltas. El problema de la identificación pretende establecer si las estimaciones numéricas de los parámetros de una ecuación estructural (ecuación 3.3) pueden ser obtenidos de los coeficientes estimados de la forma reducida (ecuación 3.5). Si puede hacerse para una ecuación, se dice que dicha ecuación está **identificada**, y en caso contrario la ecuación considerada está **subidentificada**.

Para poder calcular la matriz Π en la expresión 3.5 es necesario que el número de ecuaciones planteadas $(N \cdot d)$ sea mayor o igual que el número de incógnitas a calcular $(N \cdot K)$, que no son otras que los valores de la matriz Π, con lo que se deduce que es necesario que $K \leq d$ para que se pueda calcular la expresión reducida de un M.E.S.

Para calcular la matriz B y la matriz Γ en la expresión 3.6 se tienen NK ecuaciones, y las incógnitas a calcular serán los valores de B distintos de cero (como máximo $N(N-1)$) y los valores de Γ distintos de cero (como máximo NK). Por lo tanto, una ecuación podrá ser resuelta si el número de igualdades que aporta (K) es mayor o igual que el número de incógnitas que tiene $(k_i + n_i - 1)$.

Hay tres tipos de ecuaciones: **subidentificadas** (no se pueden resolver), **sobreidentificadas** (las soluciones de los coeficientes de la forma estructural 3.1 no se obtienen de forma única de la forma reducida 3.5) y **exactamente identificadas** (solución única).

Para estudiar la identificación de una ecuación se usan dos condiciones, la **condición de orden** y la **condición de rango**. La condición de orden es una mera comprobación cuya computación es mínima pero es una condición necesaria aunque no suficiente. La condición de rango, que sí es necesaria y suficiente, tiene necesidades computacionales grandes.

Condición de Orden

Si la ecuación i-ésima cumple $n_i - 1 > K - k_i$ es una ecuación subidentificada, si cumple $n_i - 1 = K - k_i$ es una ecuación exactamente identificada, y si cumple $n_i - 1 < K - k_i$ es una ecuación sobreidentificada [25].

Condición de Rango

En un modelo que contiene N ecuaciones y N variables endógenas, una ecuación está identificada sí y solo sí puede construirse al menos una matriz no singular de tamaño $(N-1)\times(N-1)$ con los coeficientes de las variables (endógenas y exógenas) excluidas de esa ecuación particular e incluidas en otras del modelo [25].

3.1.3. Tipos de estimadores

En el estudio teórico de los M.E.S. se han desarrollado diferentes expresiones de estimadores, cada uno bajo diferentes hipótesis y con diferentes propiedades en la inferencia del parámetro buscado. También dichos estimadores tienen diferentes necesidades de computación y de memoria. Ejemplos de ellos son el estimador de Máxima Verosimilitud (MV), Mínimos Cuadrados Indirectos (MCI), Mínimos en dos Etapas (MC2E), Mínimos Cuadrados en tres Etapas (MC3E), etc. [24, 25]

Tradicionalmente se suelen agrupar estos estimadores en dos conjuntos diferenciados: métodos de información completa y métodos de información limitada. Los primeros calculan la estimación de todas las ecuaciones del sistema a la vez, es decir, operan y resuelven todas las ecuaciones al mismo tiempo. Estos métodos gozan de una gran precisión en su estimación puesto que reflejan la influencia de unas ecuaciones en otras. En contrapartida son métodos muy costosos tanto en memoria como computacionalmente, debido al tamaño de las matrices que deben manejar, y además son muy sensibles a posibles errores. Es decir, si una ecuación está mal expresada o tiene errores en una variable, su influencia será notoria en el sistema completo sesgando gravemente las estimaciones obtenidas, incluso en ecuaciones donde no aparezca dicha variable. Estas dos desventajas hicieron que desde casi el principio quedaran en desuso a favor de los métodos de información limitada. El método más común de este tipo es MC3E.

Los métodos de información limitada, por el contrario, operan ecuación a ecuación, estimando los coeficientes en cada una de ellas, utilizando la información de la ecuación mas una información global que se ha calculado a partir del sistema. Por lo tanto, este tipo de estimador es menos preciso que los anteriores, pero menos vulnerable a errores y con menos necesidades computacionales. Estas razones llevaron en los años 60 y 70 (años donde el uso de los M.E.S. gigantes llegó a su apogeo) a utilizar básicamente estas técnicas [18], y hasta en algunos casos, y aún sabiendo el error que cometían, a usar Mínimos Cuadrados Ordinarios directamente por no poder computar estimadores más complejos. Los métodos más comunes en este grupo son MCI (que aún siendo el más simple no se puede usar siempre) y MC2E.

3.1.4. Mínimos Cuadrados en Dos Etapas (MC2E)

El estimador MC2E, a diferencia de otros estimadores usados en Modelos de Ecuaciones Simultáneas, puede ser utilizado en cualquier ecuación del M.E.S. que esté identificada, sin importar si es exactamente identificada o sobreidentificada.

Según la condición de orden, para resolver una ecuación por MC2E se tiene que cumplir que $n_i - 1 + k_i \leq K$.

Sabemos que no es posible usar MCO directamente en un Modelo de Ecuaciones Simultáneas debido a la correlación de las variables endógenas con el término aleatorio. MC2E lo que hace es sustituir las variables endógenas que actúan como variables explicativas en todas las ecuaciones del sistema por otras variables que, pareciéndose mucho a las endógenas originales, no están correlacionadas con el termino de error. A estas variables se les denomina *proxy* (por ser muy próximas a las variables endógenas originales). Una vez hecha la sustitución, se puede usar MCO para resolver las ecuaciones.

La variable *proxy* de una endógena es calculada mediante la estimación de dicha endógena utilizando MCO, con todas las exógenas del sistema como variables explicativas. Debido a que una variable endógena puede aparecer en varias ecuaciones del M.E.S. es necesario almacenar las *proxy* calculadas para su reutilización. Por lo tanto, para el cálculo de las variables proxy se aproxima $B = -I$ y se resuelve la ecuación resultante $Y = X\Gamma^T + u$ por MCO o lo que es lo mismo, se obtiene $\hat{\Gamma}^T = \min_{\Gamma^T} ||X\Gamma^T - Y||_2$. A la matriz de variables *proxy* se la denota por $\hat{Y}$, y tiene las mismas dimensiones que Y $(d \times N)$. Su expresión queda de la siguiente forma:

$$\hat{Y} = X\hat{\Gamma}^T = X(X^TX)^{-1}X^TY \tag{3.7}$$

Una vez calculadas las variables *proxy* procedemos a la resolución de las ecuaciones. En cada ecuación es necesario estimar los coeficientes de las variables endógenas y exógenas. Para la ecuación i-ésima, dichos coeficientes son los elementos no nulos de la fila i-ésima de las matriz B y de la matriz Γ. Por comodidad, denotaremos por b_i y Γ_i a los vectores formados por dichos coeficientes no nulos. Por lo tanto b_i será un vector de dimensión $n_i - 1$ y Γ_i un vector de dimensión k_i. También, por simplificar, se denotará por X_i, Y_i e $\hat{Y}_i$ a las matrices asociadas a la ecuación i-ésima que tienen por columnas los datos de las variables exógenas, endógenas y *proxy* respectivamente, y que aparecen en dicha ecuación, es decir, X_i es una matriz de dimensión $d \times k_i$, y las matrices Y_i e $\hat{Y}_i$ son de dimensión $d \times (n_i - 1)$.

Para resolver la ecuación i-ésima, la cual se puede expresar de la forma $y_i = X_ib_i + Y_i\Gamma_i + \epsilon_i$ (siendo la variable y_i la endógena principal de la ecuación), primero se sustituyen las variables endógenas que hay en dicha ecuación (salvo la principal) por las variables *proxy* calculadas (Y_i por $\hat{Y}_i$).

Definimos la matriz $X_{e_i} = [X_i|\hat{Y}_i]$, es decir, la matriz formada por las columnas de las variables exógenas y las *proxy* que aparecen en la ecuación. Por lo tanto, para la estimación de $\beta_i = [b_i^T|\Gamma_i^T]$ basta con calcular la expresión $(X_{e_i}^TX_{e_i})^{-1}X_{e_i}^Ty_i$ (expresión de MCO aplicada a X_{e_i}).

El algoritmo 1 muestra un esquema para el estimador MC2E. Como se observa, se aplican dos veces MCO para resolver cada una de las ecuaciones del sistema, una en el cálculo de las variables *proxy*, y la otra en la resolución de cada una de las ecuaciones. De ahí el nombre del estimador, Mínimos Cuadrados en Dos Etapas.

Algoritmo 1 Algoritmo básico para MC2E

Entrada: $X \in \mathbb{R}^{d\times K}, Y \in \mathbb{R}^{d\times N}, B \in \mathbb{R}^{N\times N}$ y $\Gamma \in \mathbb{R}^{N\times K}$
Salida: $B \in \mathbb{R}^{N\times N}, \Gamma \in \mathbb{R}^{N\times K}$

1: Aproximar $B = -Id$ y estimar $\hat{\Gamma}$ mediante la expresión $\hat{\Gamma}^T = (X^T X)^{-1} X^T Y$
2: Estimar $\hat{Y}$ mediante la expresión $\hat{Y} = X\hat{\Gamma}^T$
3: Sustituir Y por $\hat{Y}$ en las endógenas explicativas de la expresión 3.4 resultando $Y^T = \tilde{B}\hat{Y}^T + \Gamma X^T + u^T$
4: **Para** $i = 1...N$ **Hacer**
5: Hacer $X_{e_i} = [X_i|\hat{Y}_i]$ {Usando B_i y Γ_i}
6: $\beta_i = (X_{e_i}^T X_{e_i})^{-1} X_{e_i}^T y_i$
7: Sustituir los valores de β_i en las incógnitas de B_i y Γ_i
8: **Fin Para**

3.2. Conclusiones

En este capítulo se han descrito matemáticamente los Modelos de Ecuaciones Simultáneas desde sus expresiones más comunes (forma estructural y forma reducida), y también los tipos de ecuaciones que lo forman (problema de la identificación).

La parte final del capítulo se ha centrado en describir matemáticamente el estimador de M.E.S. que será utilizado en los algoritmos del presente trabajo, MC2E, perteneciente al grupo de estimadores llamados de información limitada.

Capítulo 4

Algoritmos QR para la resolución de M.E.S.

En este capítulo se han desarrollado algoritmos para la resolución de Modelos de Ecuaciones Simultáneas mediante el estimador MC2E utilizando la descomposición QR expuesta en la sección 2.2.2.

Se han desarrollado algoritmos utilizando las reflexiones de Householder y los rotaciones de Givens, y se han comparado sus tiempos de ejecución. En la versión basada en rotaciones de Givens se ha reutilizado la descomposición QR utilizada para el cálculo de las variables *proxy* en la resolución de las ecuaciones, reduciéndose así considerablemente el tiempo de ejecución y consiguiéndose un algoritmo eficiente para M.E.S. con grandes tamaños de muestra (su tiempo de ejecución se ve mínimamente afectado por d).

La principal aportación del capítulo es la utilización de la descomposición QR (utilizada hasta ahora en muchas técnicas basadas en Mínimos Cuadrados) en Modelos de Ecuaciones Simultáneas. Hasta donde sabemos, no habían sido usados con anterioridad en la resolución de M.E.S. desarrollos basados en reflexiones de Householder y reflectores de Givens, aunque sí se han aplicado en otras técnicas (como la búsqueda del mejor modelo en Mínimos Cuadrados [21, 28], el cómputo de todos los modelos posibles en Mínimos Cuadrados [37], etc.)

4.1. Mínimos Cuadrados en dos Etapas mediante reflexiones de Householder

Utilizando la descomposición QR expuesta en la sección 2.2.2 se pueden rediseñar los cálculos en la expresión de MC2E para conseguir reducir el coste de operaciones y a la vez reducir la inestabilidad numérica. Esta descomposición ha sido usada con anterioridad en otras técnicas econométricas basadas (como éstas) en problemas de mínimos cuadrados [20, 21, 37]. Desarrollamos ahora la misma idea en la resolución de Modelos de Ecuaciones Simultáneas.

Dada la matriz de variables exógenas X (de dimensión $d \times K$), existe una matriz ortogonal Q (de dimensión $d \times d$) y una matriz triangular R (de dimensión $d \times K$), tal que $X = QR$ o equivalentemente $Q^T X = R$. La matriz R tiene la forma $\begin{pmatrix} R_1 \\ 0 \end{pmatrix} \begin{matrix} K \times K \\ (d-K) \times K \end{matrix}$, donde R_1 es una matriz triangular superior.

Si usamos la descomposición QR en la expresión de la matriz de variables *proxy* $\hat{Y}$ dada en la ecuación 3.7, el estimador queda de la siguiente forma:

$$\begin{gathered} \hat{Y} = X(X^TX)^{-1}X^TY = QR(R^TQ^TQR)^{-1}R^TQ^TY = \\ QR\left([R_1^T|0]\begin{pmatrix} R_1 \\ 0 \end{pmatrix}\right)^{-1}[R_1^T|0]Q^TY = Q\begin{pmatrix} R_1 \\ 0 \end{pmatrix}(R_1^{-1}R_1^{-T})[R_1^T|0]Q^TY = \\ Q\begin{pmatrix} Id_K \\ 0 \end{pmatrix}[Id_K|0]Q^TY = Q\begin{pmatrix} Id_K & 0 \\ 0 & 0 \end{pmatrix}Q^TY \end{gathered} \tag{4.1}$$

Si en la expresión anterior llamamos $\tilde{Y}$ a la matriz resultante de multiplicar Q^T por Y, y si particionamos dicha matriz en dos tal que $\tilde{Y} = \begin{pmatrix} \tilde{Y}_1 \\ \tilde{Y}_2 \end{pmatrix}$ siendo $\tilde{Y}_1$ de dimensión $K \times N$, e $\tilde{Y}_2$ de dimensión $(d-K) \times N$, la expresión 4.1 queda:

$$\hat{Y} = Q\begin{pmatrix} Id_K & 0 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} \tilde{Y}_1 \\ \tilde{Y}_2 \end{pmatrix} = Q\begin{pmatrix} \tilde{Y}_1 \\ 0 \end{pmatrix} \tag{4.2}$$

Si se usa la descomposición QR en un algoritmo para MC2E, el primer paso sería la descomposición QR de X y a continuación el cálculo de la matriz de variables *proxy*, $\hat{Y}$.

De la misma manera se puede usar la descomposición QR en la estimación de los coeficientes en cada una de las ecuaciones del M.E.S. pero, en este caso, descomponiendo la matriz $X_{e_i} = Q_i R_i$, tal y como se muestra a continuación:

$$\begin{gathered} \hat{\beta}_i = (X_{e_i}^TX_{e_i})^{-1}X_{e_i}^Ty_i = (R_i^TQ_i^TQ_iR_i)^{-1}R_i^TQ_i^Ty_i = \\ \left([R_{i,1}^T|0]\begin{pmatrix} R_{i,1} \\ 0 \end{pmatrix}\right)^{-1}[R_{i,1}^T|0]Q_i^Ty_i = (R_{i,1}^{-1}R_{i,1}^{-T})[R_{i,1}^T|0]Q_i^Ty_i = \\ R_{i,1}^{-1}[Id_K|0]Q_i^Ty_i = [R_{i,1}^{-1}|0]Q_i^Ty_i = [R_{i,1}^{-1}|0]\tilde{y}_i = R_{i,1}^{-1}\tilde{y}_{i,1} \end{gathered} \tag{4.3}$$

siendo $\tilde{y}_{i,1}$ la submatriz de $\tilde{y}_i$ formada por las $n_i + k_i - 1$ primeras filas, donde $\tilde{y}_i$ es la matriz resultante de multiplicar $Q_i^Ty_i$. $R_{i,1}$ es la submatriz de R_i formada por las k_i primeras filas.

En las ecuaciones 4.1 y 4.2 es necesaria la multiplicación de la matriz Q (o su transpuesta) por otra matriz. La construcción de dicha matriz a partir de los reflectores de Householder es costosa tanto en tiempo de computación como en necesidades de memoria, por lo que es recomendable evitarlo. En lugar de construir la matriz Q y multiplicarla por otra matriz procederemos a aplicar directamente a esta última matriz los reflectores de Householder.

También hay que tener en cuenta que R_1 es una matriz triangular superior, y el coste a la hora de resolver el sistema de ecuaciones donde aparece la inversa de esta matriz es menor que en el caso de tener una matriz completa.

Tal y como se planteó al comienzo de la presente memoria se pretende hacer una aplicación lo más portable posible, y se usará siempre que sea posible librerías LAPACK y BLAS. En este caso pueden ser utilizadas tanto para la descomposición QR como en la resolución de un sistema cuya matriz de coeficientes es triangular. En los experimentos mostrados en 4.4 se han usado $dtrsv$ para la resolver el sistema (inversa de R_1), $dgeqrf$ para la descomposición QR y $dormqr$ para aplicar los reflectores ya calculados a una matriz.

El coste de aplicar $dgeqrf$ a la matriz X $(d \times K)$ es $\frac{2}{3}K^2(3d-K)$, el coste de aplicar $dorgqr$ usando K reflectores a una matriz $d \times N$ es $2NK(2d-K)$, y el coste de aplicar $dtrsv$ para resolver un sistema triangular superior de orden $n \times n$ sería n^2 [22].

El algoritmo 2 muestra el esquema de MC2E usando la descomposición QR. Tal y como se ha explicado anteriormente, en las líneas 2, 3 y 6 del algoritmo no se multiplica la matriz Q por la otra matriz, sino que se aplican los reflectores de Householder directamente. En el paso 3 del algoritmo no es necesario aplicar los reflectores a la matriz entera puesto que hay ceros por debajo de la fila K-ésima. En el paso 7 se resuelve un sistema de ecuaciones triangular superior. En las líneas 1 y 5 la descomposición QR usada es la descomposición de Householder descrita en la sección 2.2.2.

Algoritmo 2 Algoritmo $MC2E_H$

Entrada: $X \in \mathbb{R}^{d\times K}, Y \in \mathbb{R}^{d\times N}, B \in \mathbb{R}^{N\times N}$ y $\Gamma \in \mathbb{R}^{N\times K}$
Salida: $B \in \mathbb{R}^{N\times N}, \Gamma \in \mathbb{R}^{N\times K}$

1: Obtener Q y R (desc. QR de$X$) $\{\text{coste} \rightarrow \frac{4}{3}K^2(3d-K)\}$
2: $\tilde{Y} = Q^T Y$ $\{\text{coste} \rightarrow 2NK(2d-K)\}$
3: $\hat{Y} = Q \begin{pmatrix} \tilde{Y}_1 \\ 0 \end{pmatrix}$ siguiendo la ecuación 4.2 $\{\text{coste} \rightarrow 2NK^2\}$
4: **Para** i=1...N **Hacer**
5: Obtener Q_i y R_i (desc. QR de X_{e_i}) $\{\text{coste} \rightarrow \frac{4}{3}(n_i+k_i-1)^2(3d-(n_i+k_i-1))\}$
6: $\tilde{y}_i = Q_i^T y_i$ $\{\text{coste} \rightarrow 2(n_i+k_i-1)(2d-(n_i+k_i-1))\}$
7: $\beta_i = R_{i,1}^{-1}\tilde{y}_{i,1}$ $\{\text{coste} \rightarrow (n_i+k_i-1)^2\}$
8: Sustituir los valores de β en las incógnitas de B_i y Γ_i
9: **Fin Para**

En la ecuación i-ésima, la matriz X_{e_i}, formada por las variables exógenas y las variables *proxy* de la ecuación que se esté resolviendo, es la que se descompone en Q_i y R_i, e y_i es la endógena principal de dicha ecuación.

El coste total de $MC2E_H$ es:

$$T_{MC2E_H}(N,d,K,\Omega_n,\Omega_k) = \tfrac{4}{3}K^2(3d-K) + 4NKd + \sum_{i=1}^{N}\left(\tfrac{4}{3}(n_i+k_i-1)^2(3d-(n_i+k_i-1)) + 2(n_i+k_i-1)(2d-(n_i+k_i-1)) + (n_i+k_i-1)^2\right) \quad (4.4)$$

donde $\Omega_k = (k_1, ..., k_N)$ es un vector de dimensión N cuyos elementos representan el número de exógenas que hay en cada una de las ecuaciones del sistema.

Se puede observar en la expresión 4.4 que el parámetro d tiene presencia en los términos previos al sumatorio y también en los de dentro del sumatorio. Esto se traduce en un aumento del coste computacional considerable cuando aumenta el tamaño de la muestra, por lo que $MC2E_H$ no es eficiente para un M.E.S. con tamaño de muestra grande.

4.2. Mínimos Cuadrados en dos Etapas mediante rotaciones de Givens

Se detecta una gran deficiencia en la versión presentada anteriormente de descomposición QR. Ésta reside en el no aprovechamiento de la descomposición QR de X en la descomposición en cada ecuación. También puede verse como deficiente el cálculo de la matriz $\hat{Y}$, puesto que exige aplicar dos veces los reflectores de Householder a la matriz Y.

La idea general del nuevo algoritmo consiste en aprovechar en cada ecuación que la matriz X ha sido ya triangularizada para, en lugar de usar Householder sobre la matriz X_{e_i}, aplicar rotaciones de Givens haciendo cero los elementos no nulos por debajo de la diagonal principal. Es decir, no tener que volver a usar Householder en cada ecuación y, en su lugar, obtener la descomposición QR a partir de la descomposición hecha de X para el cálculo de $\hat{Y}$.

Para resolver la ecuación i-ésima se construye la matriz X_{e_i}, la cual está formada por k_i columnas de X y $n_i - 1$ columnas de $\hat{Y}$ (que han sustituido a las variables endógenas de la ecuación salvo la principal), y a continuación se hace su descomposición QR con el correspondiente ahorro en el cálculo de MCO, aplicando lo expuesto en la ecuación 4.1 pero tomando en lugar de X, X_{e_i}, e y_i en lugar de Y. Es necesario por lo tanto construir la matriz $[X|\hat{Y}]$ a partir de la cual se obtendrán todas las X_{e_i}.

Sean Q y R tal que $Q^T X = R$, donde Q y R son calculadas mediante el método de Householder. Si aplicamos Q^T a $[X|\hat{Y}]$ se obtiene (usando la expresión de $\hat{Y}$ en la ecuación 4.2) $Q^T[X|\hat{Y}]=[Q^TX|Q^T\hat{Y}] = [R|Q^T\hat{Y}] = \begin{pmatrix} R_1 & \tilde{Y}_1 \\ 0 & 0 \end{pmatrix} \begin{matrix} K \\ d-K \end{matrix}$, con R_1 triangular superior.

Por lo tanto se tiene una matriz $[R_1|\tilde{Y}_1]$ formada por las primeras K filas de la matriz, donde aparecen los valores distintos de cero. El resto de la matriz no es relevante por ser todo valores nulos.

Esto tiene consecuencias importantes puesto que ya no es necesario calcular $\hat{Y}$, sino que basta con aplicar los vectores de Householder calculados en la descomposición QR de X a Y una sola vez, y a continuación quedarse con las K primeras filas de la matriz resultante.

Nos disponemos ahora a resolver cada una de las ecuaciones del modelo. Para resolver la ecuación i-ésima, la cual se puede expresar de la forma $y_i = X_i b_i + Y_i \Gamma_i + \epsilon_i$, definimos la matriz $X_{e_i} = [X_i|\hat{Y}_i]$, es decir la matriz formada por las columnas de las variables exógenas y las variables *proxy* que aparecen en la ecuación. La matriz se puede expresar de la forma $X_{e_i} = [X|\hat{Y}]S_i$ donde S_i (de dimensión $(K+N) \times (k_i + n_i - 1)$) es una matriz de selección formada por todo ceros salvo un 1 en cada columna, que estará en la fila correspondiente a la columna que se desee seleccionar. Con lo que S_i puede ser expresada como $(e_{\lambda_{i,1}}, ..., e_{\lambda_{i,n_i+k_i-1}})$, donde $e_{\lambda_{i,j}}$ es el $\lambda_{i,j}$-ésimo vector de la matriz identidad Id_{K+N}, $\forall j = 1, ..., n_i + k_i - 1$, siendo $\lambda_{i,j}$ la columna de $[X|\hat{Y}]$ correspondiente a la variable que entra en el j-ésimo lugar en la ecuación i-ésima (por lo que aparecerá en la j-ésima columna de X_{e_i}).

Si aplicamos la matriz Q^T a X_{e_i} se obtiene que $Q^T X_{e_i} = Q^T[X|\hat{Y}]S_i = [R|Q^T\hat{Y}]S_i$ $\begin{pmatrix} R_1 & \tilde{Y}_1 \\ 0 & 0 \end{pmatrix} S_i = \begin{pmatrix} R_{i,1} & \tilde{Y}_{i,1} \\ 0 & 0 \end{pmatrix}$, siendo $R_{i,1}$ e $\tilde{Y}_{i,1}$ las K primeras filas de la matriz resultante de la multiplicación. Hay que notar que $R_{i,1}$ tendrá k_i columnas de R_1 y que no necesariamente tiene por que ser triangular superior (solo ocurrirá esto en el caso en que dicha matriz esté formada por las k_i primeras columnas de R_1). La matriz $\tilde{Y}_{i,1}$ estará formada por columnas de $\tilde{Y}_1$ todas con K filas.

Es necesario obtener la descomposición QR de X_{e_i}, por lo que hay que encontrar una matriz ortogonal que multiplicada a la anterior dé como resultado una matriz triangular superior. Para esto utilizaremos las rotaciones de Givens descritas en el apartado 2.2.2.

Supongamos que queremos retriangularizar X_{e_i} cuya matriz de selección es $S_i = (e_{\lambda_{i,1}}, ..., e_{\lambda_{i,n_i+k_i-1}})$. El número de rotaciones de Givens necesarias para triangularizar $[R_{i,1}|\tilde{Y}_{i,1}]$ es $\sum_{j=1}^{k_i}(\lambda_{i,j} - j)$ para las primeras k_i columnas y $\sum_{j=1}^{n_i-1}(K - j - k_i)$ para el resto. La matriz ortogonal obtenida a partir de multiplicar rotaciones de Givens sería:

$$\tilde{Q}_i^T = \left(\prod_{n=1}^{n_i-1} \prod_{j=1}^{K-n} G^{(i)}_{K-j-k_i, K-j-k_i+1} \right) \left(\prod_{n=1}^{k_i} \prod_{j=1}^{\lambda_{i,n}-n} G^{(n)}_{\lambda_{i,n}-j, \lambda_{i,n}-j+1} \right) \tag{4.5}$$

El coste de la retriangularización de $[R_{i,1}|\tilde{Y}_{i,1}]$ es:

$$C_i(\lambda_i, k_i, n_i) = 6 \left(\sum_{j=1}^{k_i} (\lambda_{i,j} - j)(k_i + n_i - j) + \sum_{j=1}^{n_i-1} (K - j - k_i)(n_i - j) \right) \tag{4.6}$$

siendo $\lambda_i = (\lambda_{i,1}, ..., \lambda_{i,n_i+k_i-1})$.

La figura 4.1 muestra la secuencia de eliminación de elementos en el proceso de retriangularización mediante rotaciones de Givens de una matriz $X \in \mathbb{R}^{7\times5}$ siendo $\lambda_i = (1,3,4,6,7)$. La entrada i (i=1,...,6) indica el elemento reducido a cero en la i-ésima rotación, multiplicando a la izquierda por la matriz $\tilde{Q}_i^T = G_{5,6}^{(5)} G_{5,7}^{(5)} G_{4,5}^{(4)} G_{4,6}^{(4)} G_{3,4}^{(3)} G_{2,3}^{(2)}$.

$$\begin{pmatrix} \bullet & \bullet & \bullet & \bullet & \bullet \\ & \bullet & \bullet & \bullet & \bullet \\ & 1 & \bullet & \bullet & \bullet \\ & & 2 & \bullet & \bullet \\ & & & 4 & \bullet \\ & & & 3 & 6 \\ & & & & 5 \end{pmatrix}$$

Figura 4.1: Orden de anulación en la retriangularización de una matriz $X \in \mathbb{R}^{7\times5}$.

La matriz $Q_i^T = \tilde{Q}_i^T Q^T$ es una matriz ortogonal tal que $Q_i^T X_{e_i} = R_i$ con $R_i = \begin{pmatrix} R_{i,1} \\ 0 \end{pmatrix}$, siendo $R_{i,1}$ triangular superior de dimensión $(n_i + k_i - 1) \times (n_i + k_i - 1)$.

Para que una ecuación pueda ser resuelta tiene que estar identificada y por lo tanto se tiene que dar que $n_i + k_i - 1 \leq K$ (condición de orden) o, lo que es lo mismo, el número de columnas que tomará una ecuación de $[R_1|\tilde{Y}_1]$ es a lo sumo K. Esto asegura que la matriz triangular resultante será invertible puesto que tendrá a lo sumo K filas (con más filas tendría obligatoriamente filas formadas por ceros y la matriz no sería invertible, no pudiendo ser resuelta la ecuación).

Para calcular el valor de $\hat{\beta}_i$ tal y como se ve en su expresión en la ecuación 4.3, es necesario resolver el sistema $\hat{\beta}_i = R_{i,1}^{-1} \tilde{y}_{i,1}$, siendo ${\tilde{y}_i}^T = [\tilde{y}_{i,1}^T, \tilde{y}_{i,2}^T]$ la matriz resultante de multiplicar Q_i^T por y_i, e $\tilde{y}_{i,1}$ la submatriz de $\tilde{y}_i$ formada por las $n_i + k_i - 1$ primeras filas.

Se tiene que $\tilde{y}_i = Q_i^T y_i = \tilde{Q}_i^T Q^T y_i$, y puesto que y_i es la columna i-ésima de la matriz Y, la matriz $Q^T y_i$ es la columna i-ésima de la matriz $Q^T Y$ (que denotamos $\tilde{Y}_i$) calculada anteriormente. Por lo tanto el cálculo de $\tilde{y}_i$ se reduce a multiplicar la columna i-ésima de $Q^T Y$ por $\tilde{Q}_i^T$ o, lo que es lo mismo, a aplicar las rotaciones de Givens dadas en la expresión 4.5 a $\tilde{Y}_i$.

El algoritmo 3 muestra el esquema de $MC2E_G$ (retriangularizando en cada ecuación mediante rotaciones de Givens). Tal y como se ha explicado anteriormente, en la línea 2 del algoritmo no se multiplica la matriz Q por la otra matriz sino que se aplican los reflectores de Householder directamente a esta última. En la línea 6 se aplican las rotaciones de Givens calculadas en la línea 5 a la i-ésima columna de $\tilde{Y}$ denotada por $\tilde{Y}_i$, que ha sido calculada en el paso 2. En la línea 7 se resuelve un sistema de ecuaciones triangular superior.

El coste total del algoritmo $MC2E_G$ es:

Algoritmo 3 Algoritmo $MC2E_G$

Entrada: $X \in \mathbb{R}^{d\times K}, Y \in \mathbb{R}^{d\times N}, B \in \mathbb{R}^{N\times N}$ y $\Gamma \in \mathbb{R}^{N\times K}$
Salida: $B \in \mathbb{R}^{N\times N}, \Gamma \in \mathbb{R}^{N\times K}$

1: Obtener Q y R (desc. QR de Householder de X) {coste $\rightarrow \frac{4}{3}K^2(3d-K)$}
2: $\tilde{Y} = Q^T Y$ {coste $\rightarrow 2NK(2d-K)$}
3: Crear la matriz $[R_1|\tilde{Y}_1]$
4: **Para** i=1...N **Hacer**
5: Retriangularizar la matriz $[R_{i,1}|\tilde{Y}_{i,1}]$ mediante las rotaciones de Givens dadas en la expresión 4.5 {coste $\rightarrow C_i(\lambda_i, k_i, n_i)$}
6: $\tilde{y}_i = \tilde{Q}_i^T \tilde{Y}_i$ {aplicando las rotaciones de Givens calculadas en el paso anterior} {coste $\rightarrow 6\left(\sum_{j=1}^{k_i}(\lambda_{i,j}-j) + \sum_{j=1}^{n_i-1}(K-j-k_i)\right)$}
7: $\hat{\beta}_i = R_{i,1}^{-1}\tilde{y}_{i,1}$ {coste $\rightarrow (n_i+k_i-1)^2$}
8: Sustituir los valores de β_i en las incógnitas de B_i y Γ_i
9: **Fin Para**

$$T_{MC2E_G}(N,d,K,\Omega_n,\Omega_k) = \tfrac{4}{3}K^2(3d-K) + 2NK(2d-K) + \sum_{i=1}^{N}\left(C_i(\lambda_i,k_i,n_i) + 6\left(\sum_{j=1}^{k_i}(\lambda_{i,j}-j) + \sum_{j=1}^{n_i-1}(K-j-k_i)\right) + (n_i+k_i-1)^2\right) \tag{4.7}$$

Se puede observar en la expresión que el parámetro d no tiene presencia dentro del sumatorio, por lo que se reduce considerablemente su influencia respecto a la expresión 4.4, haciendo de $MC2E_G$ un algoritmo eficiente para M.E.S. con tamaño de muestra grande.

4.3. Paralelización del algoritmo $MC2E_G$

Se presentan en esta sección la versión en paralelo del algoritmo 3 desarrollado en secuencial en la sección anterior. La paralelización está diseñada para memoria compartida aunque el cambio a memoria distribuida es casi inmediato.

No se han desarrollado una verión en paralelo del algoritmo $MC2E_H$ porque se entiende que será usado en su lugar $MC2E_G$ que mejora el tiempo de ejecución para cualquier tamaño del problema (ver tabla 4.1).

El algoritmo 4 muestra un esquema paralelo del algoritmo $MC2E_G$ para p procesadores con memoria compartida.

En las líneas 1 y 2 se pueden usar las funciones de LAPACK *dgeqrf* y *dorgqr* dado que están desarrolladas en numerosas versiones de LAPACK en paralelo (en memoria compartida). A continuación se hace un reparto cíclico de las ecuaciones entre los p procesadores. El hecho de hacerlo cíclico es porque en muchas ocasiones un sistema real se forma a partir de varios más pequeños concentrando las ecuaciones más grandes, que se añaden para interrelacionar las existentes, de forma consecutiva.

Algoritmo 4 Algoritmo $PMC2E_G$

Entrada: $X \in \mathbb{R}^{d\times K}$, $Y \in \mathbb{R}^{d\times N}$, $B \in \mathbb{R}^{N\times N}$ y $\Gamma \in \mathbb{R}^{N\times K}$
Salida: $B \in \mathbb{R}^{N\times N}$ y $\Gamma \in \mathbb{R}^{N\times K}$

1: Obtener Q y R (desc. QR de Householder de X en paralelo) {coste $\rightarrow \frac{4K^2}{3p}(3d-K)$}
2: Calcular $\tilde{Y} = Q^T Y$ {Aplicar los reflectores de Householder en paralelo a la matriz} {coste $\rightarrow \frac{2NK}{p}(2d-K)$}
3: Crear la matriz $[R_1|\tilde{Y}_1]$
4: EN PARALELO cada procesador $q = 0, ..., p-1$ HACE
5: **Para** j=1...$\frac{N}{p}$ **Hacer**
6: $i = q + (j-1)p + 1$
7: Retriangularizar la matriz $[R_{i,1}|\tilde{Y}_{i,1}]$ mediante las rotaciones de Givens dadas en la ecuación 4.5 {coste $\rightarrow C_i(\lambda_i, k_i, n_i)$}
8: Calcular $\tilde{y}_i = \tilde{Q}_i^T \tilde{Y}_i$ {aplicando las rotaciones de Givens calculadas en el paso anterior} {coste $\rightarrow 6\left(\sum_{j=1}^{k_i}(\lambda_{i,j}-j) + \sum_{j=1}^{n_i-1}(K-j-k_i)\right)$}
9: Calcular $\hat{\beta}_i = R_{i,1}^{-1}\tilde{y}_{i,1}$ {coste $\rightarrow (n_i+k_i-1)^2$}
10: Sustituir los valores de β_i en las incógnitas de B_i y Γ_i
11: **Fin Para**
12: FIN PARALELO

El coste total del algoritmo $PMC2E_G$ es:

$$T_{PMC2E_G}(N,d,K,\Omega_n,\Omega_k,p) = \tfrac{4}{3}\tfrac{K^2}{p}(3d-K) + 2\tfrac{N}{p}K(2d-K) + \max_{q=0...p-1}\left\{\sum_{s=1,\ i=q+(s-1)p+1}^{\frac{N}{p}}\left(C_i(\lambda_i,k_i,n_i) + 6\left(\sum_{j=1}^{k_i}(\lambda_{i,j}-j) + \sum_{j=1}^{n_i-1}(K-j-k_i)\right) + (n_i+k_i-1)^2\right)\right\} \approx \tfrac{4}{3}\tfrac{K^2}{p}(3d-K) + 2\tfrac{N}{p}K(2d-K) + \sum_{i=1}^{\frac{N}{p}}\left(C_i(\lambda_i,k_i,n_i) + 6\left(\sum_{j=1}^{k_i}(\lambda_{i,j}-j) + \sum_{j=1}^{n_i-1}(K-j-k_i)\right) + (n_i+k_i-1)^2\right) \tag{4.8}$$

Se ha considerado la siguiente aproximación: El tiempo total del algoritmo es el tiempo del último procesador que termine, por lo que es necesario tomar el máximo variando los procesadores $q = 0, ...p-1$. Sin embargo, si se hace una asignación suficientemente balanceada se asume que todos los procesadores acaban simultáneamente, por lo que se puede tomar el tiempo de uno de ellos como aproximación.

4.4. Estudio experimental

En esta sección se van a describir diversos experimentos mediante los cuales se estudian los algoritmos propuestos en los apartados anteriores. Los Modelos de

Ecuaciones Simultáneas han sido generados de dos formas diferentes.

Se han generado M.E.S. aleatorios sin restringir el número de variables por ecuación pero obligando a que todas sean identificadas. El procedimiento es el siguiente: Se generan aleatoriamente las N ecuaciones del M.E.S. tomando aleatoriamente $n_i - 1$ valores entre 1 y N (serán las variables endógenas) además del valor i (será la endógena principal), y k_i valores entre 1 y K (serán las variables exógenas) con la condición de que la ecuación sea identificada. Una vez creado el modelo hay que generar los datos que sigan la relación impuesta por el modelo. Por ello se generan aleatoriamente K variables con d datos cada una, las cuales formarán la matriz de variables exógenas X. A continuación se calculan las N variables endógenas (matriz Y) siguiendo la ecuación 3.5 con la variable de ruido blanco v generada siguiendo una $N(0, \sigma)$, con σ=0.01.

Se pretenden estudiar experimentalmente los siguientes puntos:

- La influencia de los parámetros N, K y d en $MC2E_H$ y $MC2E_G$.
- Comparar los tiempos de ejecución de $MC2E_H$ y $MC2E_G$ para diferentes tamaños del problema.

La tabla 4.1 muestra una comparativa entre los tiempos de ejecución de la resolución de un M.E.S. usando los algoritmos $MC2E_H$ y $MC2E_G$. Se han tomado para cada tamaño del problema 5 modelos que han sido resueltos por ambos algoritmos, anotando los tiempos de ejecución y el ratio. A continuación se han calculado los promedios y las desviaciones típicas tanto de los tiempos como de los ratios, mostrándolos en la tabla. El hecho de tomar promedios ha sido para evitar que el tiempo de ejecución quede sesgado por el M.E.S. generado aleatoriamente. A priori, puede parecer que el algoritmo $MC2E_G$ tiene una gran dependencia respecto al M.E.S. que se está resolviendo. Esto es debido a que el coste de las retriangularizaciones depende de las variables que se hayan tomado en cada ecuación. Sin embargo, una vez vistos los resultados, se puede concluir que la variabilidad es mínima y que tal dependencia no afecta considerablemente al tiempo total del algoritmo.

Se puede observar que el tiempo de resolución de un sistema utilizando retriangulaciones de Givens es siempre menor que el tiempo utilizado por el algoritmo que usa la descomposición de Householder en cada ecuación. Esta diferencia se hace especialmente efectiva cuando el tamaño de la muestra d es muy grande con respecto al número de endógenas y exógenas. El ratio mayor se da en el caso en que mayor diferencia hay entre N, K y d ($N = 100$, $K = 100$ y $d = 1000$) y el caso menor en el que menor diferencia hay ($N = 200$, $K = 400$ y $d = 500$). Puesto que la condición $K \leq d$ pone un límite a K, se podría concluir que el algoritmo $MC2E_G$ (algoritmo 3) es más eficiente en general que $MC2E_H$.

El algoritmo $MC2E_G$ no se ve afectado prácticamente por el aumento de d, pero su tiempo de ejecución aumenta considerablemente con el aumento de N y K. También es de subrayar la poca variabilidad a la que se ven afectados ambos algoritmos (y por supuesto el ratio). Esto índica que dichos algoritmos se ven afectados

Tam. del problema			Tiempo	Tiempo	
N	K	d	Householder	Givens	Ratio
100	100	500	$0.50_{0.02}$	$0.15_{0.00}$	$3.35_{0.06}$
100	100	1000	$1.06_{0.03}$	$0.16_{0.00}$	$6.63_{0.21}$
100	200	500	$1.39_{0.06}$	$0.67_{0.02}$	$2.09_{0.09}$
100	200	1000	$3.23_{0.11}$	$0.70_{0.00}$	$4.60_{0.13}$
200	200	500	$3.39_{0.08}$	$1.90_{0.03}$	$1.79_{0.01}$
200	200	1000	$7.85_{0.32}$	$1.94_{0.05}$	$4.05_{0.07}$
200	400	500	$10.25_{0.14}$	$9.16_{0.13}$	$1.12_{0.02}$
200	400	1000	$23.28_{0.20}$	$9.31_{0.16}$	$2.50_{0.05}$
400	400	1000	$56.60_{2.47}$	$28.05_{0.41}$	$2.02_{0.07}$
400	400	1500	$98.85_{2.84}$	$28.62_{0.31}$	$3.45_{0.08}$
400	600	1000	$108.68_{3.94}$	$74.55_{1.18}$	$1.46_{0.03}$
400	600	1500	$200.76_{5.57}$	$75.13_{0.91}$	$2.67_{0.05}$
800	800	2000	$1319.30_{12.58}$	$438.14_{2.24}$	$3.01_{0.01}$
800	800	2500	$1889.01_{20.66}$	$440.61_{3.57}$	$4.29_{0.02}$
800	1000	2000	$2067.94_{16.62}$	$741.28_{4.60}$	$2.79_{0.01}$
800	1000	2500	$2894.90_{28.64}$	$741.77_{5.43}$	$3.90_{0.02}$

Tabla 4.1: Tiempo de ejecución (en segundos) y ratio de los algoritmos $MC2E_H$ y $MC2E_G$ en Rosebud, cuando se varía el número de variables endógenas (N), el número de variables exógenas (K) y el tamaño de la muestra (d). Se muestra el promedio (número grande) y la desviación típica (subíndice) de 5 medidas para el mismo tamaño del problema.

mínimamente por la estructura del sistema en cuanto a tiempo de ejecución, siendo dicho tiempo muy similar para todos los sistemas con el mismo número de variables endógenas y exógenas y mismo tamaño de muestra.

La tabla 4.2 muestra los tiempos de ejecución (en segundos) y el speed-up del algoritmo $PMC2E_G$. Puesto que la paralelización del algoritmo es relativamente sencilla por la independencia que se tiene entre ecuaciones, los speed-up resultantes son bastante altos incluso para tamaños del problema pequeños.

4.5. Conclusiones

Se han desarrollado algoritmos del estimador MC2E utilizando las reflexiones de Householder y las rotaciones de Givens. Se han comparado los tiempos de ejecución tanto teóricamente como experimentalmente. En la comparación de ambos algoritmos, se observa que el algoritmo basado en rotaciones de Givens tiene un coste menor que el basado en las reflexiones de Householder.

Con $MC2E_G$ se obtiene un nuevo algoritmo para el estimador MC2E que re-

N	K	d	1 proc.	2 proc.	Sp	4 proc.	Sp	8 proc.	Sp
400	400	1000	27.87	14.22	1.96	7.18	3.88	3.71	7.51
400	400	1500	28.03	14.33	1.96	7.25	3.87	3.74	7.49
400	600	1000	75.82	37.90	2.00	19.36	3.92	10.01	7.57
400	600	1500	76.40	38.01	2.01	19.48	3.92	10.08	7.58
800	800	2000	435.11	218.16	1.99	112.75	3.86	57.42	7.58
800	800	2500	439.88	220.03	2.00	113.72	3.87	58.12	7.57
800	1000	2000	741.75	370.05	2.00	188.08	3.94	96.36	7.70
800	1000	2500	736.48	368.26	2.00	187.07	3.94	95.62	7.70
1000	1000	2500	1058.59	531.54	1.99	266.22	3.98	135.76	7.80
1000	1200	3000	1634.33	816.77	2.00	418.91	3.90	210.10	7.78
1200	1200	2500	2191.15	1095.92	2.00	552.06	3.97	281.44	7.79
1200	1200	3000	2176.38	1092.76	1.99	550.75	3.95	279.97	7.77

Tabla 4.2: Tiempos de ejecución (en segundos) en Rosebud y Speed-up correspondiente al algoritmo $PMC2E_G$, cuando se varía el número de variables endógenas (N), el número de variables exógenas (K) y el tamaño de la muestra (d).

duce considerablemente el tiempo de ejecución, tiene la estabilidad numérica de los algoritmos basados en la QR (se evitan inversas, multiplicaciones grandes, etc.), y además con su versión paralela se obtienen speed-ups satisfactorios.

Capítulo 5

Algoritmos QR con nodos ficticios para la resolución de M.E.S.

En el capítulo anterior se ha explotado la propiedad de que la matriz de datos de cada ecuación está formada por columnas de la matriz X (de la cual ha sido calculada su descomposición QR) y la matriz Y. En lugar de volver a realizar la descomposición QR en cada ecuación se retriangulariza usando rotaciones de Givens, siendo el ahorro computacional muy notable. Para reducir aún más el número de operaciones, y por lo tanto el tiempo de ejecución, tendremos en cuenta dos considerariones.

La primera es la de no resolver las ecuaciones en cualquier orden, puesto que se podría dar el caso de que una ecuación contuviera todas las variables de otra y se pudiera retriangularizar a partir de la matriz asociada a la primera (y por lo tanto aprovechar los cálculos hechos para ella) obteniéndose la matriz asociada a la segunda. Sin embargo, se verá en este capítulo que solo en muy raras ocasiones se puede dar en un M.E.S. una ecuación identificada, y por lo tanto que se pueda resolver, cuyas variables, salvo su endógena principal, estén todas incluidas en otra ecuación.

La segunda consideración consiste en observar que la retriangularización expuesta en el capítulo anterior tiene mucho más coste cuanto más diferentes son las matrices correspondientes a cada ecuación, por lo que la creación de matrices intermedias podría reducir el número de retriangularizaciones a usar para hallar la descomposición QR en cada ecuación. En este capítulo, se explota esta idea mediante la creación de un árbol cuyos nodos corresponden a conjuntos de variables. En unos casos corresponderán a las ecuaciones del M.E.S. y en otros casos (los nodos llamados ficticios) no corresponden a ninguna ecuación del M.E.S. pero contribuyen a reducir considerablemente el número de rotaciones de Givens a utilizar para obtener las descomposiciones QR en cada ecuación.

Puesto que la finalidad de la construcción del árbol no es otra que la de recortar la computación requerida a la hora de resolver las ecuaciones del sistema, es importante que el tiempo de obtención del árbol más el tiempo de la resolución de las ecuaciones utilizando el árbol no sea superior al tiempo de resolver las ecuaciones

retriangularizando directamente. Puesto que el número de posibles nodos en el árbol podría ser muy grande y el número de caminos o ejes entre ellos también, se hace impracticable el uso de algoritmos exhaustivos, siendo la solución propuesta en su lugar un algoritmo heurístico.

La principal aportación del capítulo es la utilización de nodos ficticios que permitan reducir el número de rotaciones de Givens en la descomposición QR de cada ecuación. Esta idea ha sido utilizada en la generación y resolución de un subconjunto de modelos de todos los posibles modelos en Mínimos Cuadrados a partir de un conjunto de variables [37]. Hasta donde sabemos, no se han usado nodos ficticios y árboles de mínimo coste en la resolución de M.E.S.

5.1. Orden de resolución de las ecuaciones de un Modelo de Ecuaciones Simultáneas

Tal y como se ha dicho al comienzo del capítulo, lo idóneo sería resolver las ecuaciones en un orden de forma que si la ecuación i-ésima tiene todas sus variables incluidas en la ecuación j-ésima, se resolviera primero la segunda y después la primera. Esto permitiría obtener la descomposición QR de la matriz asociada a la ecuación i-ésima a partir de la descomposición de la matriz asociada a la ecuación j-ésima, ahorrando en el número de rotaciones de Givens respecto a retriangularizar desde la descomposición QR de la matriz $[X|\hat{Y}]$.

Sin embargo, el orden de resolución es prácticamente indiferente puesto que solo en muy raras ocasiones se puede dar en un M.E.S. una ecuación identificada cuyas variables, salvo su endógena principal, estén todas incluidas en otra ecuación.

De hecho no se puede dar una ecuación identificada cuyas variables estén todas (endógena principal incluida) en otra ecuación. Esta propiedad se puede deducir de la condición de rango (sección 3.1.2), la cual dice que una ecuación de un M.E.S. con N variables endógenas está identificada sí y solo sí existe al menos una matriz de dimensión $(N-1)\times(N-1)$ de coeficientes de variables excluidas en dicha ecuación con determinante distinto de cero. Supongamos que la ecuación i-ésima tiene todas sus variables en la ecuación j-ésima. Cuando se construye la matriz de tamaño $(N-1)\times(N-1)$ para comprobar la condición de rango en la j-ésima ecuación, la fila correspondiente a la ecuación i-ésima contiene todo ceros (puesto que hay que poner solo coeficientes excluidos) por lo que el determinante es cero.

Por lo tanto, la única posibilidad de que se dé una ecuación con todas las variables explicativas en otra es que la endógena principal no esté contenida en ella. Un ejemplo podría ser el siguiente:

$$\begin{aligned} y_1 &= \beta_{1,3}y_3 + \gamma_{1,2}x_2 + \gamma_{1,3}x_3 + u_1 \\ y_2 &= \beta_{2,3}y_3 + \gamma_{2,3}x_3 + u_2 \\ y_3 &= \beta_{3,1}y_1 + \gamma_{3,1}x_1 + u_3 \end{aligned} \tag{5.1}$$

donde la ecuación 2 tiene todas sus variables explicativas en la ecuación 1, y por

lo tanto si resolvemos primero la ecuación 1 se puede retriangularizar su matriz llegando a la ecuación 2 con menos coste. El problema es que estas ecuaciones no son comunes puesto que tiene que darse la propiedad de que no estando la endógena en la otra ecuación sí lo estén todas las variables explicativas (situadas a la derecha de la igualdad) lo que no tiene mucho sentido estadístico. En el ejemplo se incluyen las variables y_3 y x_3 en la primera ecuación, y por lo tanto se admite que ambas influyen en y_1. Sin embargo no se incluye y_2, admitiendo en este caso que y_2 no influye en y_1. Luego resulta que hay una ecuación que dice que y_2 se puede expresar en función de y_3 y x_3 más un error. No obstante, si se diera el caso de una ecuación de este tipo es claro que la reducción en el coste computacional en ella sería grande.

5.2. Árbol de Mínimo Coste

Tal y como se ha explicando al comienzo del capítulo, surge el problema de obtener el árbol de mínimo coste que permita obtener la descomposición QR de la matriz asociada a cada ecuación usando matrices intermedias. Dichas matrices (tanto las asociadas a las ecuaciones como las intermedias) formarán los nodos del árbol. En esta sección se formaliza el problema y se desarrollan las reglas necesarias para construir un algoritmo que obtenga un árbol que se aproxime al de mínimo coste.

Definición *Árbol asociado a un M.E.S.*.
Definimos el árbol asociado a un M.E.S. como un conjunto de nodos y aristas donde los nodos representan conjuntos de variables del sistema tanto exógenas como endógenas. Las aristas representan una relación de contenido o subconjunto de variables. Si existe una arista de un nodo N_i a otro nodo N_j indica que todas las variables de N_j se encuentran en N_i y, por lo tanto, se puede llegar de este último nodo al primero y obtener los datos de sus variables.□

Un nodo tiene asociada una matriz formada por un conjunto de columnas de la matriz $R = \begin{pmatrix} R_1 & \tilde{Y}_1 \\ 0 & 0 \end{pmatrix} \begin{matrix} K \\ d-K \end{matrix}$ definida en el capítulo anterior (y que será la matriz asociada al nodo raíz o nodo cero) retriangularizadas. Es decir, se calcula la descomposición QR de la matriz resultante de seleccionar las columnas correspondientes a las variables del nodo, y la R resultante es la matriz asociada a dicho nodo.

Sea M_i la matriz asociada al nodo N_i, el cual suponemos que tiene q variables. Tenemos que:

- Si $q \leq K$ entonces M_i es triangular superior de dimensión $d \times q$.
- Si $K \leq q \leq K+N$, entonces $M_i = \begin{pmatrix} R_{i,1} & \tilde{Y}_{i,1} \\ 0 & 0 \end{pmatrix} \begin{matrix} K \\ d-K \end{matrix}$ siendo $R_{i,1} \in \mathbb{R}^{K \times K}$ triangular superior y $\tilde{Y}_{i,1} \in \mathbb{R}^{K \times (q-K)}$.

El pasar de un nodo a otro en el árbol a través de las aristas existentes da como resultado en las matrices asociadas la eliminación de las columnas del primer nodo que no están en el segundo, y por lo tanto es necesaria la retriangularización hasta conseguir la matriz asociada al nodo. La retriangularización se realizará usando rotaciones de Givens, y el número de operaciones a realizar depende en gran medida del camino por el que se llegue al nodo. Se realizarán menos operaciones cuantas menos columnas hayan sido eliminadas del nodo origen al nodo destino o, dicho de otra forma, cuanto más se parezcan ambos. Puesto que tendremos varios caminos para llegar al mismo nodo surge el problema de decidir cual de ellos utilizar (el que menos coste en número de operaciones tenga) y por lo tanto la eliminación de los otros, por lo que de todos los posibles grafos resultará un árbol cuyo coste sea el mínimo posible.

A los nodos correspondientes a las ecuaciones los llamaremos nodos ecuación y a los nodos que no correspondan a ninguna ecuación los llamaremos *nodos ficticios*. Supondremos que en total (sumando nodos ficticios y nodos ecuación) hay G nodos más el raíz.

Representaremos las variables de un nodo por su número si son variables exógenas y por su número con gorro si son endógenas.

Ejemplo N_i=[1 2 5 $\hat{2}$ $\hat{4}$] es un nodo asociado a la matriz cuyas columnas corresponden a las variables exógenas 1, 2 y 5 y a las variables endógenas 2 y 4, una vez que han sido sustituidas por las variables *proxy*.

Definición Sea v una variable, decimos que $v \in N_i$ si la matriz asociada a N_i tiene a v como una de sus columnas.□

Definición *Nivel del nodo.*
Decimos que q es el nivel de un nodo N_i, o que $N_i \in L_q$, si el número de variables o columnas de la matriz asociada a N_i es q.□

Nota No existe N_i con $i = 0, ..., G$ tal que $N_i \in L_1$.

El nodo raíz del árbol tendrá asociada una matriz con todas las variables posibles y su nivel será $N + K$, es decir, $M_0 = \begin{pmatrix} R_1 & \tilde{Y}_1 \\ 0 & 0 \end{pmatrix} \begin{matrix} K \\ d-K \end{matrix} \in L_{N+K}$.

En general, en un nivel q puede existir un número de nodos $C_q^{N+K} = \frac{(N+K)!}{q!(N+K-q)!}$, y por lo tanto en total en el árbol pueden existir un máximo de $|V|_{max} = \sum_{i=1}^{N+K} C_i^{N+K}$ $= \sum_{i=0}^{N+K} C_i^{N+K} - 1 = 2^{N+K} - 1$ nodos.

Desde el nivel q el número máximo de aristas que se pueden construir desde nodos de ese nivel a nodos inferiores es $C_q^{N+K}(2^{N+K-q} - 2)$, y por lo tanto en total en el árbol pueden existir un máximo de $|A|_{max} = \sum_{i=0}^{N+K-2} C_i^{N+K}(2^{N+K-i} - 2) =$ $3^{N+K} - 2^{N+K+1} + 1$ aristas.

Este último cálculo está basado en la igualdad $\sum_{i=0}^{n} C_i^n C_{k-i}^{n-i} t^i = C_k^n (1+t)^k$ para cualesquiera n, k y t números enteros (que se puede encontrar en la página 76 de [19]). Tomando $k = n$ y teniendo presente que $C_i^n = C_{n-i}^n$, se obtiene la propiedad de que $\sum_{i=0}^{n} C_{n-i}^n t^i = (1+t)^n$, y llamando $j = n - i$ se obtiene que $\sum_{j=0}^{n} C_j^n t^{n-j} = (1+t)^n$, que es usada en el cálculo de la expresión anterior.

Definición *Distancia Asimétrica.*
Definimos la distancia asimétrica de un nodo N_i a otro N_j ($d_i(N_j)$) como el número de variables que están en N_i y no están en N_j:

$$d_i(N_j) = |\{x_k \in N_i \ con \ k = 1, ..., q \ y \ x_k \notin N_j \ siendo \ q \ / \ N_i \in L_q\}| \ .\square$$

Nota $d_0(N_i) = 0, \ \forall \ i = 1, ..., G.$

Nota Si existe un arco de N_j a N_i entonces $d_i(N_j) = 0$.

Definición *Distancia Total.*
Definimos la distancia total (d_{ij}) de dos nodos como la suma de las distancias asimétricas entre ellos: $d_{ji} = d_{ij} = d_i(N_j) + d_j(N_i).\square$

Definición *Coste de un arco.*
El coste del arco $a_{i,j}$ que va desde el nodo N_i al nodo N_j se representa por $C_{i,j}$, y es el número de operaciones necesarias para conseguir la matriz asociada al nodo N_j a partir de la matriz asociada al nodo N_i mediante retriangulaciones de Givens (ver sección 2.2.2).$\square$

El coste de un arco desde el nodo raíz hasta un nodo N_i viene dado por la siguiente expresión, siendo n_i en número de variables endogenas y k_i el número de exógenas de la ecuación i-ésima:

$$C_{0,i} = \sum_{j=1, \ m_j \in N_i}^{k_i} (m_j - j)(n_i + k_i - j + 1) + \sum_{j=k_i+1}^{n_i+k_i} (K - j)(n_i + k_i - j + 1) \quad (5.2)$$

siendo $m_j \in N_i$ la variable j-ésima del nodo N_i (cuya posición en el nodo raíz sería m_j).

Para triangularizar las columnas tomadas de la submatriz R_1 de M_0 (primer sumatorio), se necesitan $m_j - j$ eliminaciones para la columna j-ésima. En cada una de ellas se utiliza una rotación de Givens que actualiza dicho elemento y los $n_i + k_i - j + 1$ que hay hasta el final de la matriz. Para triangularizar las columnas tomadas de la submatriz $\tilde{Y}_1$ de M_0 (segundo sumatorio), se necesitan siempre $K - j$ eliminaciones puesto que $\tilde{Y}_1$ tiene K filas, y el número de elementos hasta el final son $n_i + k_i - j + 1$, puesto que j toma valores a partir de $k_i + 1$.

Y en el caso general, el coste de un arco desde el nodo N_s hasta un nodo N_i viene dado por la expresión:

$$\begin{aligned} C_{s,i} &= \sum_{j=1,\ m_j \in N_i}^{k_i+n_i} (L(m_j, N_s) - j)(n_i + k_i - j + 1) \ si\ K > L(m_{k_i+n_i}, N_s) \\ C_{s,i} &= \sum_{j=1,\ m_j \in N_i}^{j_c} (L(m_j, N_s) - j)(n_i + k_i - j + 1) + \sum_{j=j_c+1}^{k_i+n_i} (K - j)(n_i + k_i - j + 1) \\ &si\ K \leq L(m_{k_i+n_i}, N_s) \end{aligned} \tag{5.3}$$

donde j_c es el valor para el que $L(m_{j_c}, N_s) = K$, donde $L(m_j, N_s)$ es el lugar que ocupa m_j en N_s.

En el cálculo de 5.3 se utiliza el mismo razonamiento que para la expresión 5.2, teniendo en cuenta que m_j, variable que ocupa la columna j-ésima en M_i y que por lo tanto debe ser triangularizada hasta tener ceros por debajo del elemento j-ésimo, ya no procede de una matriz donde ocupaba la columna m_j-ésima (como ocurría en M_0), sino que en M_s ocupa la posición $L(m_j, N_s)$. En este caso la variable m_j tendrá $L(m_j, N_s)$ valores distintos de cero y el número de elementos a eliminar será $L(m_j, N_s) - j$, y en cada uno de ellos habrá que actualizar hasta el final de la matriz.

Puede ocurrir que M_s sea una matriz triangular, o que la última variable de N_i, tome una posición en N_s menor que K. En ambos casos (el primero es un caso particular del segundo) las variables a triangularizar tendrán un número de elementos distintos de cero igual a su posición, y no sería necesario eliminar elementos desde la fila K (como ocurría en $C_{0,i}$). Si por el contrario hubiera un valor de j (que llamaremos j_c) a partir del cual las variables que se incluyen en N_i tienen posiciones en N_s superiores a K, sería necesario incluir un segundo sumatorio que contara las operaciones de eliminar desde el elemento K hasta el j (como se observa en la segunda expresión para $C_{s,i}$).

Nota Si existe $a_{s,i}$, entonces $m_j \geq L(m_j, N_s) \geq j$. Esto es debido a que la posición de la variable j-ésima en un nodo N_i, es menor o igual que la que ocupa en el nodo N_s, la cual es a su vez menor que la que ocupa en el nodo raíz.

Definición *Árbol de Mínimo Coste.*
Definimos el Árbol de Mínimo Coste (AMC) de un M.E.S. como el árbol, de entre todos los asociados al M.E.S., que minimiza la suma total de los costes de todos los arcos del árbol.□

5.3. Construcción del Árbol de Mínimo Coste

En un principio se debe formar un árbol colocando los nodos correspondientes a cada una de las ecuaciones y el nodo raíz, y añadir los arcos que unen este último

nodo con el resto. Cualquier árbol que se plantee con nodos ficticios tiene que tener los nodos del árbol descrito dentro de él. De esta forma lo único que variará serán los nodos ficticios añadidos y los arcos entre dichos nodos y los nodos ya existentes. El fin será que la suma del coste total de dichos arcos sea lo menor posible.

Un algoritmo que obtenga el mínimo absoluto pasa por un alto coste computacional. Sería necesario, por cada combinación de nodos ficticios posibles, encontrar el árbol de mínimo coste y el AMC sería el que minimizara el coste de todos los encontrados para todas las combinaciones posibles de nodos ficticios. El alto número de nodos ficticios y de arcos posibles hace que dicho algoritmo no sea eficiente. También hay que subrayar que lo que se busca es una forma de resolver las ecuaciones que reduzca el coste de retriangularizar desde el nodo cero a todas ellas, y si el coste de encontrar dicho algoritmo es superior a resolver dichas ecuaciones directamente, no tendría sentido el aplicarlo.

Por lo tanto, se propone un algoritmo heurístico basado en una serie de reglas que se detallan a continuación, mediante el cual se pretende, si no encontrar el mínimo global, encontrar una solución que esté suficientemente cerca del mínimo y con la que se reduzca, al menos en ciertos problemas, el tiempo de computación.

Primero se detallarán y justificarán las reglas heurísticas que serán utilizadas en el algoritmo, y seguidamente se explicará el algoritmo y como se usan las reglas en él.

Regla 1 Sean N_1, N_2 y N_3 tres nodos tal que $N_1 \in L_{p_1}$, $N_2 \in L_{p_2}$ y $N_3 \in L_{p_3}$. Supongamos que existen los arcos $a_{1,3}$ y $a_{2,3}$. Si existierá $a_{1,2}$ se podría eliminar $a_{1,3}$ puesto que siempre ocurrirá que $C_{2,3} \leq C_{1,3}$ y el camino para ir a N_3 sería a través de N_2.

Ejemplo En un sistema con $N = 6$ y $K = 8$, si se tienen los nodos $N_1 = (2\ 3\ 7\ \hat{4}\ \hat{5})$ y $N_2 = (3\ 4\ 7\ \hat{5}\ \hat{6})$, el coste de retriangularizar desde N_0 es de 13 para N_1, es decir, $C_{0,1} = 13$, y 15 para N_2 ($C_{0,2} = 15$). En total el coste de ambos es de 15+13=28. Si creamos el nodo $N_3 = (2\ 3\ 4\ 7\ \hat{4}\ \hat{5}\ \hat{6})$ los costes son $C_{0,3} = 12$, $C_{3,1} = 3$ y $C_{3,2} = 7$, lo que da un total de 22 con lo que añadiendo el nodo ficticio N_3 y aplicando la regla 1 dos veces, una para los nodos N_0, N_1 y N_3 y otra para los nodos N_0,N_2 y N_3 (donde en ambos casos N_0 hace de nodo N_1 en la regla, N_3 de N_2, y el nodo restante de nodo N_2), se eliminan los caminos $a_{0,1}$ y $a_{0,2}$, y se ahorran 6 operaciones.

Regla 2 Dados dos nodos N_i y N_j, de todos los nodos ficticios N_k tal que $d_k(N_i) = d_k(N_j) = 0$ siempre se creará el que minimiza la suma de las distancias asimétricas: $\min_k\{d_i(N_k) + d_j(N_k)\}$.

La regla 2 dice que el nodo ficticio que, una vez dados dos nodos, más se parece a ambos (es decir el que más variables comunes tiene) es el nodo a tomar como camino intermedio para ir a ambos. Con esta regla se intenta que la retriangularización sea lo menos costosa posible por parecerse el nodo origen (que es el ficticio que se crea) a los dos de destino (que pueden ser nodos ficticios o ecuaciones). Pero los nodos

ficticios pueden variar según el orden de nodos dos a dos que se tomen, por lo que lo lógico es comenzar por aquellos que se parezcan más y por lo tanto la retriangulación desde el nodo nuevo que se cree sea menor.

En el ejemplo anterior, N_3 es el nodo (de todos para los que ocurre que $d_1(N_3) = d_2(N_3) = 0$) que minimiza la suma de las distancias, $d_3(N_1) + d_3(N_2)$. Esto hace que sea el más cercano a N_1 y N_2 a la vez, y que por lo tanto sea menor el número de operaciones necesarias para la retriangularización desde él.

Nota Dados $N_i \in L_q$ y $N_j \in L_{q'}$ tal que $d_i(N_j) = r$ y $d_j(N_i) = r'$, el nodo ficticio N_k creado según la regla 2 pertenecerá a $L_{q+r'} = L_{q'+r}$, es decir, N_k subirá r' niveles respecto a L_q y r niveles respecto a $L_{q'}$.

Regla 3 Para crear un nodo ficticio se seleccionan aquellos dos nodos cuya distancia total sea mínima.

En realidad esto nos dará un orden en el algoritmo de actuación, buscando siempre a la hora de crear un nuevo nodo ficticio los nodos con menor distancia total. También implica un coste de inicialización alto (orden cúbico) de comparaciones puesto que es necesario calcular las distancias entre todos los nodos, teniendo que comparar en cada cálculo las variables de los dos nodos. Sin embargo, se ha comprobado experimentalmente que, salvo para tamaños del problema pequeños, este tiempo no es significativo en comparación con el tiempo del resto del algoritmo.

Regla 4 Si existe un nodo ficticio N_i con un solo arco de salida este puede ser eliminado (puesto que no representa mejora ninguna) junto con dicho arco creando un nuevo arco desde el nodo padre al hijo.

Ejemplo Veamos con este ejemplo como la regla 4 da lugar a una reducción en el número de nodos ficticios usados. Denotaremos un nodo ficticio cuyos descendientes sean N_i y N_j por $NF_{i,j}$ En el mismo sistema anterior con $N = 6$ y $K = 8$, si se tienen los nodos $N_1 = (2\ 3\ 4\ 7\ \hat{4}\ \hat{5}\ \hat{6}) \in L_7$, $N_2 = (2\ 3\ 4\ 6\ \hat{4}\ \hat{5}\ \hat{6}) \in L_7$ y $N_3 = (2\ 3\ 4\ 8\ \hat{4}\ \hat{5}\ \hat{6}) \in L_7$, ocurre que $d_1(N_2) = d_1(N_3) = d_2(N_1) = d_3(N_1) = d_2(N_3) = d_3(N_2) = 1$. Como todas las distancias son iguales podemos crear cualquier nodo ficticio. Los nodos ficticios que se pueden crear son $NF_{1,2} = (2\ 3\ 4\ 6\ 7\ \hat{4}\ \hat{5}\ \hat{6}) \in L_8$, $NF_{1,3} = (2\ 3\ 4\ 7\ 8\ \hat{4}\ \hat{5}\ \hat{6}) \in L_8$ y $NF_{2,3} = (2\ 3\ 4\ 6\ 8\ \hat{4}\ \hat{5}\ \hat{6}) \in L_8$. Por ejemplo, a N_2 se puede llegar tanto desde $NF_{1,2}$ (cuyo coste es de 10) como desde $NF_{2,3}$ (cuyo coste es de 11). Como el coste de llegar a N_2 desde ambos es 3, se crea $NF_{1,2}$ ($NF_{2,3}$ no se crea siguiendo la regla 4). Lo mismo se hace con $NF_{1,3}$ cuyo coste es 12, siendo los costes desde él a N_1 y N_3 3 y 4, respectivamente. Por lo tanto, a N_1 se irá desde $NF_{1,2}$, y $NF_{1,3}$ no se crea por la regla 4.

Para la siguiente regla se necesitan dos definiciones adicionales:

Definición $N_i \in L_q$ está más a la izquierda que $N_j \in L_q$ si existe $1 \leq j_0 \leq q$ tal que $x_k = x'_k$, $\forall\ k = 1, ..., j_0 - 1$ y $x_{j_0} < x'_{j_0}$, siendo $x_k \in N_i$ y $x'_k \in N_j\ \forall\ k = 1, ..., j_0$.□

Definición Definimos los descendientes de un nodo N_i como el conjunto $Desc(N_i)$= $\{N_j$ con $j = 0, ..., G$ y $j \neq i$ / $d_j(N_i) = 0\}$.□

Regla 5 En caso de igualdad de distancia entre n nodos del mismo nivel, se deben colocar de izquierda a derecha según el orden dado en la definición anterior y crear $\lfloor \frac{n}{2} \rfloor$ nodos ficticios de forma que cada uno de ellos tenga dos de los n nodos como descendientes. En caso de igual distancia entre nodos de diferente nivel, se añadirán los nodos ficticios por orden de coste (del arco a este nodo ficticio) de menor a mayor.

En el ejemplo anterior el orden es N_2, N_1, N_3 puesto que todos coinciden en las tres primeras variables y difieren en la cuarta. Por lo tanto, según la regla 5 se debería crear el nodo ficticio entre N_2 y N_1, es decir $NF_{1,2}$ sin necesidad de comprobar los costes.

También en el ejemplo anterior puede darse el caso de que exista un nodo $N_4 =$ (6 8 $\hat{6}$) cuyo coste es $C_{0,4} = 16$. Al decidir incorporar $NF_{1,2}$ en lugar de $NF_{2,3}$ se ganaba en una operación, pero en este (existiendo N_4) caso, si se hubiera incorporado $NF_{2,3}$ se podría haber colocado N_4 como descendiente suyo (puesto que $d_4(NF_{2,3}) = 0$) obteniéndose N_4 con coste 11. Por lo tanto, haber incorporado $NF_{2,3}$ en lugar de $NF_{1,2}$ habría costado 4 operaciones menos en el coste total del árbol. Para intentar paliar en gran parte este problema se enuncia la regla 6.

Regla 6 En caso de igualdad de distancia entre nodos, se deben crear los nodos ficticios que tengan mayor número de descendientes.

La regla 6 servirá como primer criterio para crear nodos ficticios habiendo igualdad de distancia entre nodos, y si se diera igualdad también en esta regla pasaríamos a usar la 5.

Regla 7 Si $N_k, N_j \in Desc(N_i)$ y $N_i \in L_s$, un nodo ficticio $NF \in L_q$ cuyos descendientes sean N_k, N_j no se creará si $q \geq s$.

En la figura 5.1 se muestra el árbol de resolución de un M.E.S. con 5 variables endógenas y 8 exógenas. Las variables endógenas han sido representadas en cada nodo por su número más K, también se les coloca un gorro para su distinción y para indicar que han sido sustituidas por las variables *proxy*. El nodo N_0 está formado por todas las variables exógenas y todas las endógenas. El acceso a cada una de las ecuaciones se hace desde el nodo cero retriangularizando la matriz resultante por rotaciones de Givens.

En la figura 5.2 se representa el mismo árbol que en la figura 5.1, pero se han añadido nodos ficticios (representados por rectángulos) siguiendo las reglas descritas hasta ahora y aplicadas en el orden que describe el algoritmo que se verá en la siguiente sección. El primer nodo ficticio en ser añadido ha sido N_6 ya que, según la regla 3 hay que añadir nodos ficticios entre nodos con distancia mínima. Puesto que hay igualdad en la distancia menor ($d_{1,2} = d_{3,4}$) se aplica la regla 5. Después se

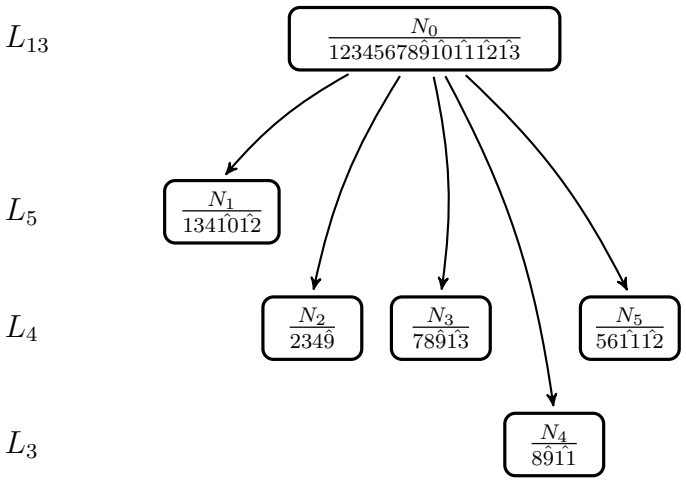

Figura 5.1: Árbol de resolución de un M.E.S. con N=5 y K=8 donde se representan los nodos ecuaciones y el nodo raíz por niveles (mostrados a la izquierda). En cada nodo se representan las variables que lo componen, siendo las que llevan gorro las variables endógenas y las que no las exógenas.

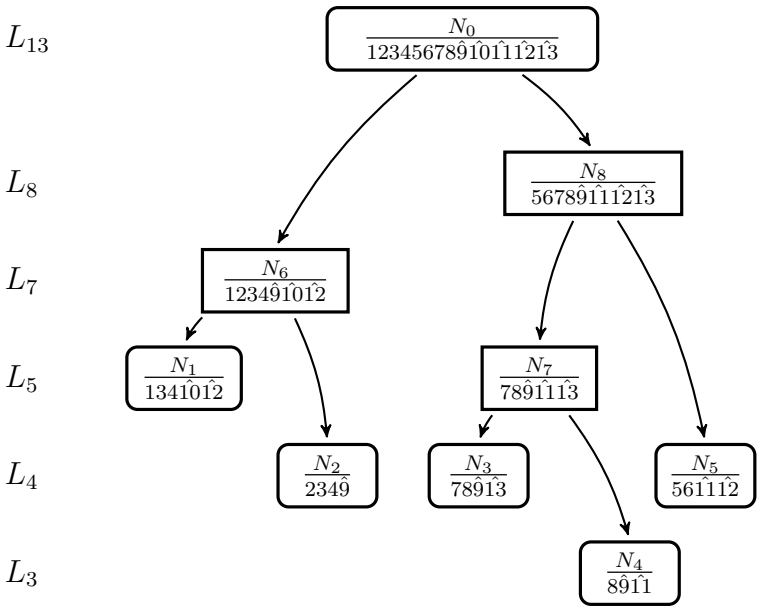

Figura 5.2: Árbol de resolución del M.E.S. dado en la figura 5.1 donde se han añadido tres nodos ficticios.

han añadido los nodos N_7 y N_8 en este orden siguiendo también la regla 3. No se ha añadido un nodo ficticio que tuviera como descendientes a N_7 y N_8 por coincidir con N_0 e incumplir la regla 7.

Regla 8 Un nodo ficticio se añade al árbol si reduce el coste entre un nodo existente y sus hijos.

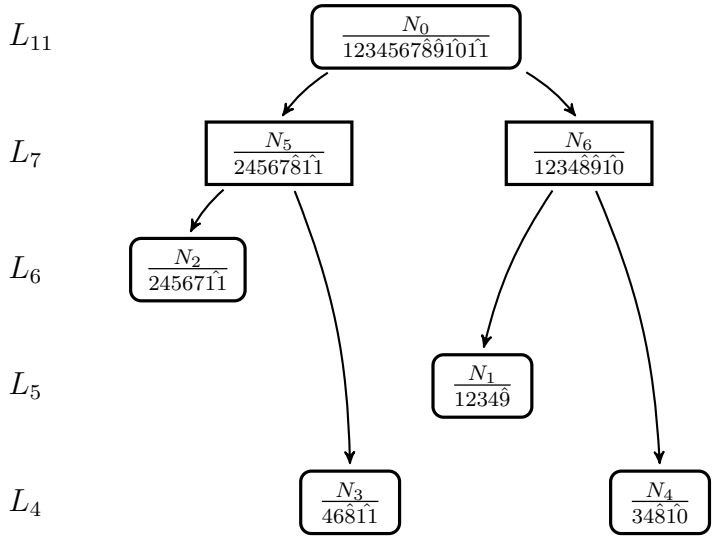

Figura 5.3: Árbol de resolución de un M.E.S. donde se han añadido dos nodos ficticios. El nodo N_5 mejora el coste global y el nodo N_6 lo empeora.

La regla 8 viene a decir que no todos los nodos reducen el coste del árbol, por lo que solo se añadirán los que lo hagan. En el árbol mostrado en la figura 5.3 se observan cuatro nodos ecuaciones y dos nodos ficticios (representados por un rectángulo), uno de los cuales se incluye (el nodo N_5) y otro no (N_6) por la regla 8. En la parte izquierda del árbol, si no se añade N_5 se tienen los costes $C_{0,3} = 35$ y $C_{0,2} = 35$, que suman 70, y al añadir el nodo N_5 se tienen $C_{0,5} = 45$, $C_{5,3} = 19$ y $C_{5,2} = 1$, lo que suma 65. Por lo tanto el añadir el nodo N_5 disminuye el coste global del árbol. En la parte derecha, si no se añade N_6 se tienen los costes $C_{0,4} = 25$ y $C_{0,1} = 2$, lo que suma 27, y al añadir el nodo N_6 los costes son $C_{0,6} = 8$, $C_{6,4} = 21$ y $C_{6,1} = 1$, que suman 30. Por lo tanto al añadir el nodo N_6 el coste global del árbol aumenta.

Se presentan a continuación el algoritmo heurístico que aproximará un Árbol de Coste Mínimo (AMC) y el algoritmo de resolución de Modelos de Ecuaciones Simultáneas que lo usa ($MC2E_{NF}$). La complejidad de éstos algoritmos es estudiada experimentalmente en la siguiente sección, no estudiándose la complejidad teórica dada su alta dependencia del problema abordado (el coste de resolver un sistema con nodos ficticios depende en gran medida de los nodos obtenidos, los cuales dependen de las ecuaciones y por lo tanto del problema que se aborde).

Algoritmo 5 Algoritmo *AMC*

Entrada: $B \in \mathbb{R}^{N\times N}$ y $\Gamma \in \mathbb{R}^{N\times K}$
Salida: AMC

Crear árbol con $N_0 = R$, N_i nodo correspondiente a la ecuación i, $\forall i = 1, ..., N$ y arcos desde N_0 a N_i
Crear tabla de distancias entre nodos
$G = N$, $i = 0$
Mientras $i \leq G$ **Hacer**
  $seguir =$ **Verdadero**
  Mientras ($|Desc(N_i)| \geq 2$) **Y** ($seguir =$ **Verdadero**) **Hacer**
    Buscar N_j, $N_k \in Desc(N_i)$ / $d_{j,k} = min\{d_{r,s}$ / $N_r, N_s \in Desc(N_i)\}$ {Regla 3}
    Si el mínimo no es único aplicar en este orden la regla 6 y la 5
    Si ($d_j(N_k) \geq q_i - q_j$) **O** ($d_k(N_j) \geq q_i - q_k$) **Entonces**
      seguir=**Falso** {Regla 7}
    Si no
      CrearNodo (N_{G+1}) {Según regla 2}
      Añadir N_{G+1} al árbol y a la tabla de distancias
      Calcular $a_{G+1,j}$ y $a_{G+1,k}$
      Eliminar $a_{i,j}$ y $a_{i,k}$ {Regla 1}
      Añadir N_j, N_k a $Desc(N_{G+1})$ y quitarlos de $Desc(N_i)$
      Mientras exista $N_{k'}$ / $d_{k'}(N_{G+1}) = 0$ **Hacer**
        Calcular $a_{G+1,k'}$
        Buscar N_s nodo tal que $N_{k'} \in Desc(N_s)$, y si $C_{s,k'} > C_{G+1,k'}$ eliminar $a_{s,k'}$
        Añadir N'_k a $Desc(N_{G+1})$ y quitarlo de $Desc(N_s)$
        Si $|Desc(N_s)| = 1$ **Entonces**
          Buscar N_a / $N_s \in Desc(N_a)$
          Si $N_d \in Desc(N_s)$ **Entonces** crear $a_{a,d}$ y añadir N_d a $Desc(N_a)$
          Borrar $a_{a,s}, a_{s,d}$ y N_s {Regla 4}
        Fin si
      Fin Mientras
      Si Se reduce el coste global del árbol **Entonces**
        $G = G + 1$
      Si no
        Quitar N_{G+1} y deshacer cambios {Regla 8}
      Fin si
    Fin si
  Fin Mientras
  $i = i + 1$
Fin Mientras

Los puntos 1 y 2 del algoritmo AMC (algoritmo 5) forman la inicialización del algoritmo y pueden ser relativamente costosos en problemas con muchas ecuaciones pues tiene un orden cúbico en número de comparaciones. La idea del algoritmo es tomar un nodo i con un número de descendientes mayor que 2, y crear nodos ficticios por cada dos de sus descendientes reduciéndolos hasta que posea a lo mucho dos (bucle interno, líneas 6-33) añadiendo cada vez que se crea un nodo ficticio todos los descendientes posibles (líneas 17-26). Esto se repite por cada nodo (bucle externo, líneas 4-35). Si un nodo queda con un único descendiente, se elimina del árbol (líneas 21-25). En la línea 28 se añade un nodo si reduce el coste global (lo que solo se puede averiguar una vez hechos los cálculos de las líneas anteriores). Si el nodo no reduce el coste se deshace lo calculado (líneas 12-20).

El algoritmo comienza con $i = 0$, es decir, añadiendo nodos ficticios para nodos descendientes de N_0. Una vez que ya no puede añadir más pasará a hacerlo a los descendientes del nodo N_1, y así hasta llegar a los nodos ficticios añadidos. Por la forma de crear dichos nodos (según la regla 2) la única posibilidad de que se puedan generar ficticios descendientes de ellos es que se les hayan añadido más descendientes en las líneas 17 a la 26. Cada vez que se añade un nodo ficticio se comprueba si se le pueden añadir más descendientes que los dos que lo generaron. Cuantos más descendientes se añadan al nodo, más rentable se hace y menor coste tendrá el árbol resultante, puesto que al añadirle un descendiente se le quita a otro nodo cuyo arco a él es más costoso. A su vez se comprueba que ese otro nodo no quede con un solo descendiente puesto que si se da dicho caso habrá que eliminarlo por la regla 4 (líneas 21-24). El algoritmo finaliza cuando no se pueden añadir más nodos.

El algoritmo no utiliza operaciones en coma flotante. Aun así su coste puede ser alto debido al gran número de comparaciones que debe realizar. En la inicialización del algoritmo (línea 2) se tiene un coste cúbico. Dentro del bucle, en la búsqueda dada en la línea 7 se tiene orden cuadrático, al añadir un nuevo nodo a la tabla de distancias (línea 13) también se tiene el mismo orden. También se tiene orden cuadrático en el cálculo de los costes de los arcos y en la búsqueda de un nodo ficticio que sea padre de uno dado. Por lo tanto, se tiene orden cuadrático dentro de dos bucles (líneas 4 y 6) con lo que el orden se incrementa hasta la cuarta en el peor de los casos.

El algoritmo 6 ($MC2E_{NF}$) muestra un esquema para el estimador MC2E con retriangularizaciones mediante reflexiones de Givens (similar al expuesto en el algoritmo 3) utilizando el Árbol de Mínimo Coste desarrollado en el punto anterior.

Al igual que antes, en la línea 2 del algoritmo $MC2E_{NF}$ se aplican los reflectores de Householder directamente a la matriz. En la línea 4 se usa el algoritmo 5 para aproximar el Árbol de Mínimo Coste.

En $MC2E_{NF}$, en cada nodo del árbol de AMC se retriangulariza la matriz mediante rotaciones de Givens (línea 8). Dichas rotaciones se aplican también a la matriz $\tilde{Y}'_{N_i}$ (si estamos en el nodo i-ésimo). Esta matriz está formada por las columnas de $\tilde{Y}$ correspondientes a las variables endógenas que están de endógenas principales en ecuaciones accesibles desde dicho nodo. La matriz $\tilde{Y}'_{N_i}$ está formada

Algoritmo 6 Algoritmo $MC2E_{NF}$

Entrada: $X \in \mathbb{R}^{d\times K}, Y \in \mathbb{R}^{d\times N}, B \in \mathbb{R}^{N\times N}$ y $\Gamma \in \mathbb{R}^{N\times K}$
Salida: $B \in \mathbb{R}^{N\times N}, \Gamma \in \mathbb{R}^{N\times K}$

1: Obtener Q y R {desc. QR de Householder de X}
2: Calcular $\tilde{Y} = Q^T Y$
3: **Si** no se ha hallado el AMC **Entonces**
4: Crear el Árbol de Mínimo Coste siguiendo el algoritmo 5
5: **Fin si**
6: **Repetir**
7: Recorrer el árbol y en cada nodo i hacer:
8: Retriangularizar la matriz $[R_{i,1}|\tilde{Y}_{i,1}]$ mediante las rotaciones de Givens
9: Crear $\tilde{Y}'_{N_i}$ submatriz de $\tilde{Y}_{N_s}$ ($N_i \in Desc(N_s)$) de variables endógenas principales cuyas ecuaciones se encuentren accesibles desde el nodo i-ésimo
10: Calcular $\tilde{Y}_{N_i} = \tilde{Q}_i^T \tilde{Y}'_{N_i}$ {aplicando las rotaciones de Givens calculadas en el paso anterior}
11: **Si** N_i es una ecuación **Entonces**
12: Calcular $\hat{\beta}_i = R_{i,1}^{-1} \tilde{Y}_{N_i,1}$ {$\tilde{Y}_{N_i,1}$ está formado por las primeras $n_i + k_i - 1$ filas de $\tilde{Y}_i$}
13: **Fin si**
14: **Hasta que** Fin del Árbol

por columnas de la matriz $\tilde{Y}$ que han sido multiplicadas por rotaciones de Givens en tantas ocasiones como número de nodos se hayan recorrido en el árbol de N_0 a N_i. En muchos casos no es necesario calcular todas las columnas de $\tilde{Y}_{N_i}$ puesto que se pueden obtener de la matriz asociada al nodo. Esto ocurre en nodos ficticios que tengan variables explicativas que son endógenas principales en ecuaciones que están por debajo de él. Cuando se llega al nodo ecuación, $\tilde{Y}_{N_i}$ será un vector columna que coincidirá con $\tilde{y}_i$ del paso 6 del algoritmo 3. En la línea 12 del algoritmo 6, se resuelve un sistema de ecuaciones triangular superior de la misma forma que ocurría en los algoritmos del capítulo anterior.

Como ejemplo del funcionamiento del algoritmo $MC2E_{NF}$ veamos como opera sobre el ejemplo dado en la figura 5.2. En las líneas 1 y 2 se obtienen la descomposición QR de la matriz X y la matriz de variables *proxy*, y por lo tanto se puede formar el nodo N_0 que tendrá de matriz asociada a $[R_{0,1}|\tilde{Y}_{0,1}]$. A continuación se crea el nodo N_6, para lo que se eliminan del nodo N_0 las columnas correspondientes a las variables $5, 6, 7, 8, \hat{11}$ y $\hat{13}$, y se retriangulariza la matriz utilizando rotaciones de Givens. Dichas rotaciones deberían ser aplicadas también a la matriz $\tilde{Y}'_{N_6}$ que es la matriz resultante de tomar las dos columnas de $\tilde{Y}_{0,1}$ correspondientes a las variables endógenas $\hat{9}$ y $\hat{10}$ (que son las dos primeras columnas de QY). Sin embargo, no es necesario aplicar las rotaciones puesto que estas columnas pueden ser copiadas de la matriz asociada al nodo N_6. El hecho de tomar estas columnas es porque por debajo del nodo N_6 están accesibles los nodos N_1 y N_2, que son nodos ecuación

cuyas endógenas principales son $\hat{9}$ y $\hat{10}$. Después de N_6 se llega a N_1, y en este nodo se eliminan las columnas de la matriz asociada a N_6 correspondientes a las variables 2 y $\hat{9}$, y se retriangulariza la matriz, aplicándose las rotaciones calculadas en dicha retriangularización a la matriz Y_{N_1} (que está formada por la primera columna de Y_{N_6}). En este caso sí es necesario hacer los cálculos puesto que no se dispone de esta variable en la matriz asociada al nodo. Como estamos en un nodo ecuación se resuelve la ecuación (línea 12 del algoritmo).

Al retroceder de nuevo al nodo N_6 se libera la memoria de las matrices creadas para N_1. El recorrido en este orden del árbol intenta minimizar en tamaño de la memoria usada puesto que para un árbol relativamente grande, el número de matrices en memoria puede ser muy considerable.

En el nodo N_2 se procede de la misma forma que en el N_1, y al retroceder hasta el N_0 se libera la memoria ocupada por las matrices de N_2 y N_6. A continuación se recorre la parte derecha del árbol siguiendo el mismo esquema.

5.4. Paralelización del algoritmo $MC2E_{NF}$

Se presenta en esta sección la versión en paralelo del algoritmo $MC2E_{NF}$ (algoritmo 6) desarrollado en secuencial anteriormente. La paralelización está diseñada para memoria compartida, pudiéndose reformular para memoria distribuida de forma sencilla.

No se ha desarrollado una verión en paralelo para el algoritmo AMC (algoritmo 5) porque no conseguiríamos mejorar en coste computacional al algoritmo $MC2E_G$ tal y como se demuestra a continuación. Para este razonamiento se usarán los datos tomados en el estudio experimental de este capítulo.

Supongamos que se consigue una versión en paralelo de AMC con la que se obtiene el speed-up óptimo teórico. Obviamente la usaríamos para los problemas dados en la tabla 5.1, en los que el tiempo del algoritmo AMC es alto. Tal y como se observa en la columna 8 de la tabla 5.1, el algoritmo $MC2E_G$ consigue tiempos 3, 4 e incluso 5 veces inferiores a la suma de tiempos dados por los algoritmos AMC y $MC2E_{NF}$. Además, este algoritmo, como se demuestra en la tabla 4.2 del capítulo anterior, tiene una paralelización muy buena, con lo que, aún consiguiendo reducir los tiempos del algoritmo AMC en la paralelización, siempre quedarían considerablemente por encima de los tiempos dados por la versión paralela de $MC2E_G$.

Por lo tanto, tan solo quedaría la posibilidad de desarrollar dicha paralelización para usarla en tamaños del problema en los que el algoritmo $MC2E_{NF}$ es más eficiente. Estos son los estudiados en la tabla 5.2. Pero en este caso los tiempos de cálculo del AMC no son muy altos debido al pequeño valor de N, y no es muy importante su paralelización. Además, tal y como se demuestra en las tablas 5.5 y 5.6, la paralelización del algoritmo $MC2E_{NF}$ se hace eficiente cuando se tienen muchos nodos ficticios en el AMC. Cuando no es el caso, pasar de dos procesadores no es muy eficiente, y es preferible la utilización de la versión paralela de $MC2E_G$.

El algoritmo 7 ($PMC2E_{NF}$) muestra el esquema paralelo del algoritmo 6 ($MC2E_{NF}$) para p procesadores con memoria compartida. Al igual que en el capítulo anterior, en las líneas 1 y 2 se usan funciones de LAPACK en paralalelo (puesto que en muchos sistemas están desarrolladas para poder ser usadas en memoria compartida). En la línea 4 se usa el algoritmo AMC para encontrar el Árbol de Mínimo Coste. En la línea 15 se crea la matriz $\tilde{Y}_{N_i}$ a partir de $\tilde{Y}_{N_s}$ con $N_i \in Desc(N_s)$ (lo cual es posible puesto que ésta última se ha construido a la vez que $[R_{j,1}|\tilde{Y}_{j,1}]$). La matriz $\tilde{Y}_{N_i}$ está formada por las variables endógenas principales cuyas ecuaciones se encuentran accesibles desde el nodo i-ésimo.

Algoritmo 7 Algoritmo $PMC2E_{NF}$

Entrada: $X \in \mathbb{R}^{d\times K}, Y \in \mathbb{R}^{d\times N}, B \in \mathbb{R}^{N\times N}$ y $\Gamma \in \mathbb{R}^{N\times K}$
Salida: $B \in \mathbb{R}^{N\times N}, \Gamma \in \mathbb{R}^{N\times K}$

1: Obtener Q y R {desc. QR de Householder de X en paralelo}
2: Calcular $\tilde{Y} = Q^T Y$ {Aplicar los reflectores de Householder en paralelo a la matriz}
3: **Si** No se ha hallado el AMC **Entonces**
4: Crear el Árbol de Mínimo Coste siguiendo el algoritmo 5
5: **Fin si**
6: EN PARALELO cada procesador $q = 0, ..., p-1$ HACE
7: **Repetir**
8: Recorrer el árbol y en cada nodo i hacer:
9: **Si** $i \equiv q \ mod \ p + 1$ **Entonces**
10: **Mientras** $[R_{j,1}|\tilde{Y}_{j,1}]$ no haya sido creada (siendo $i \in Desc(N_j)$) **Hacer**
11: Esperar
12: **Fin Mientras**
13: Crear la matriz $[R_{i,1}|\tilde{Y}_{i,1}]$ a partir de $[R_{j,1}|\tilde{Y}_{j,1}]$
14: Retriangularizar la matriz $[R_{i,1}|\tilde{Y}_{i,1}]$ mediante las rotaciones de Givens
15: Crear $\tilde{Y}'_{N_i}$ submatriz de $\tilde{Y}_{N_s}$ ($N_i \in Desc(N_s)$) de variables endógenas principales cuyas ecuaciones se encuentren accesibles desde el nodo i-ésimo
16: Calcular $\tilde{Y}_{N_i} = \tilde{Q}_i^T \tilde{Y}'_{N_{i-1}}$ {aplicando las rotaciones de Givens calculadas en el paso anterior}
17: **Si** N_i es una ecuación **Entonces**
18: Calcular $\hat{\beta}_i = R_{i,1}^{-1} \tilde{Y}_{N_i,1}$ {$\tilde{Y}_{N_i,1}$ está formado por las primeras $n_i + k_i - 1$ filas de $\tilde{Y}_i$}
19: **Fin si**
20: **Fin si**
21: **Hasta que** Recorrer todo el AMC
22: FIN PARALELO

La paralelización del algoritmo tiene una clara influencia en la forma en que se recorra el AMC, tanto en memoria como en tiempo de ejecución. A continuación se describien dos posibilidades:

$PMC2E_{NF}$ con recorrido del AMC en profundidad

Esta forma de paralelizar es muy acertada en los casos en los que el árbol es relativamente grande, puesto que en cada nodo que se resuelve es necesario guardar $[R_{i,1}|\tilde{Y}_{i,1}]$ e $\tilde{Y}_{N_i}$, lo que puede suponer una gran cantidad de datos y consecuentemente grandes necesidades de memoria. En un recorrido en profundidad, el algoritmo $PMC2E_{NF}$ sigue el mismo recorrido del árbol que el algoritmo $MC2E_{NF}$ pero resolviendo cada procesador un nodo. En este tipo de recorrido, se mantiene un menor número de datos en memoria que en otros recorridos, liberándose las matrices creadas en cada nodo en el momento en que se han procesado todos los nodos por debajo de él. El problema reside en que un nodo no puede ser resuelto si el nodo del cual desciende no ha sido resuelto previamente. Esto puede hacer que unos procesos tengan que esperar a otros. Sin embargo, si el árbol es suficientemente grande respecto al número de procesadores el tiempo de espera se reduce muchísimo.

En la figura 5.4 se muestra como opera el algoritmo $PMC2E_{NF}$ con recorrido en profundidad con 2 procesadores en el árbol de la figura 5.2. El número colocado encima de cada nodo indica el procesador que lo ha de resolver.

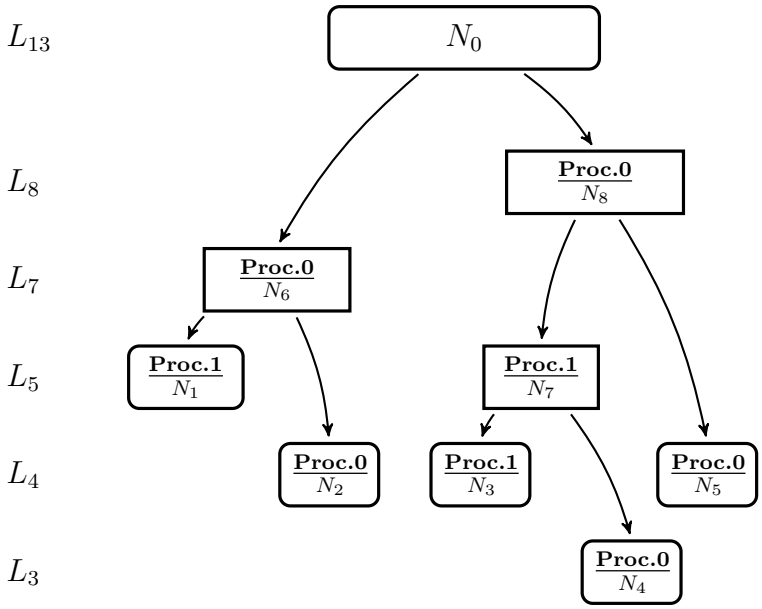

Figura 5.4: Recorrido en profundidad del algoritmo $PMC2E_{NF}$ del AMC dado en la figura 5.2.

Para la comparación del tiempo de ejecución de las paralelizaciones descritas anteriormente, simplificaremos los tiempos empleados en la resolución de cada nodo. Supondremos que se tarda 1 unidad de tiempo en hacer un nodo ficticio (retriangularizar la matriz asociada al nodo) y dos unidades de tiempo en hacer un nodo ecuación (puesto que además de retriangularizar hay que resolver un sistema), el tiempo en resolver el árbol en secuencial sería 13 unidades. En paralelo el procesador

0 hace el nodo N_6 mientras que el procesador 1 está situado en el nodo N_1 esperando. Cuando termina (llevamos por lo tanto una unidad de tiempo) el procesador 1 hace N_1 y el procesador 2 hace N_2 simultáneamente (dos unidades de tiempo). A continuación el procesador 0 hace el nodo N_8 mientras que el procesador 1 espera en N_7 (una unidad de tiempo). A continuación el procesador 1 hace el nodo N_7 mientras que el procesador 0 hace el nodo N_5. Cuando el procesador 1 termina (una unidad de tiempo), el procesador 0 todavía está a la mitad de N_5, por lo que comienza con N_3. Ahora, el procesador 0 termina una unidad de tiempo despúes de que el procesador 1 comience con N_3, por lo que hace N_4 (dos unidades de tiempo más puesto que el procesador 1 termina antes que él). En total se han usado 8 unidades de tiempo.

Sin embargo, se observa que hay una dependencia grande entre procesos (si en lugar de dos procesadores fueran más la dependencia es mayor) y parte del tiempo se pierde en la espera. No sería difícil reasignar los nodos para minimizar esta dependencia, sin embargo para árboles suficientemente grandes el tiempo de espera disminuye notablemente respecto al tiempo total.

Recorrido del árbol en amplitud

Esta forma de paralelizar es apropiada cuando el árbol es pequeño. En este caso, la cantidad de memoria no será excesivamente grande, reduciéndose el tiempo de espera considerablemente. En un recorrido en amplitud del árbol se resuelven los nodos por niveles, desde las capas más altas hasta las ecuación. Además hay que tener en cuenta que las matrices asociadas a los nodos de las capas altas son mayores que las de las capas bajas, siendo mayor el número de rotaciones de Givens necesarias para retriangularizarlas. Por ésto, el tiempo de espera de los procesos en el recorrido en profundidad se incrementa en las capas altas. En este caso, se evita dicha espera asignando procesadores a cada nodo de la capa (como se muestra en la figura 5.5). Se muestra como opera el algoritmo $PMC2E_{NF}$ con recorrido en amplitud con 2 procesadores en el árbol de la figura 5.2. De nuevo se indica encima de cada nodo el procesador que lo ha de resolver.

Suponemos, igual que en el caso anterior, que se tarda 1 unidad de tiempo en hacer un nodo ficticio y dos unidades de tiempo en hacer un nodo ecuación. El procesador 0 hace el nodo N_8 mientras que el procesador 1 hace el nodo N_6 (una unidad de tiempo). Cuando terminan, el procesador 0 es asignado a N_1 y el procesador 1 a N_7. Termina el procesador 1 (una unidad de tiempo) y el procesador 0 necesita otra unidad para terminar, por lo que se asigna a N_2. Una unidad de tiempo más tarde termina el procesador 0, que es asignado a N_3. Una unidad de tiempo después el procesador N_1 se asigna a N_5 y por último el N_0 a N_4. El tiempo total es 7, que es menor que las 8 unidades que tardaba el otro recorrido y las 13 del algoritmo secuencial.

Sin embargo, las matrices creadas en los nodos N_6, N_7 y N_8 no son eliminadas hasta el final, ocupando en todo el proceso una cantidad de memoria considerable.

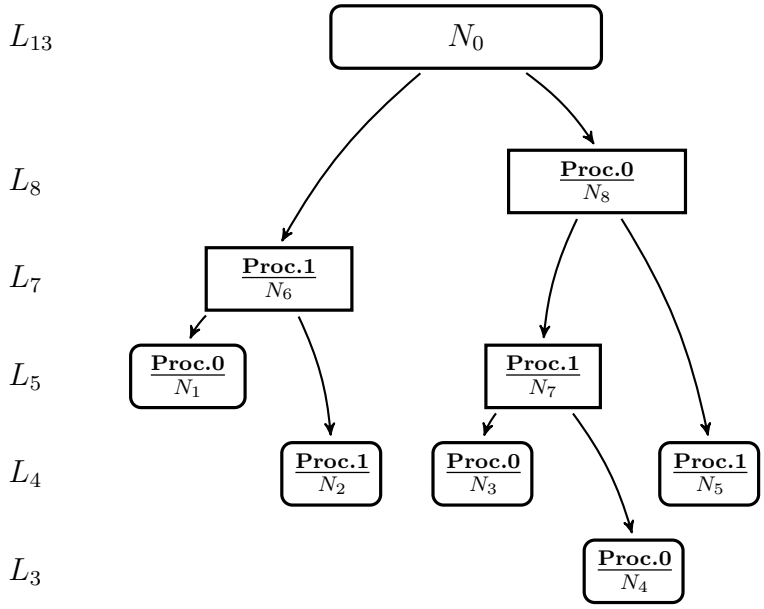

Figura 5.5: Recorrido en amplitud del algoritmo $PMC2E_{NF}$ del AMC dado en la figura 5.2.

5.5. Estudio experimental

En esta sección se van a describir diversos experimentos mediante los cuales se estudian los algoritmos propuestos en los apartados anteriores. Se han utilizado diferentes formas de generar los modelos de ecuaciones simultáneas dependiendo de los intereses que se tengan en el estudio de cada algoritmo.

Se pretenden estudiar experimentalmente los siguientes puntos:

- La influencia de los parámetros N, K y d en AMC y $MC2E_{NF}$.
- Comparar los tiempos de ejecución de $MC2E_{NF}$ y $MC2E_G$ para diferentes tamaños del problema, en los casos que se tenga creado el AMC y en los casos que se tenga que hallar llamando a AMC.
- Nodos añadidos y rechazados en la creación del árbol AMC, así como la reducción del número de operaciones en $MC2E_{NF}$ usando el árbol.
- Estudio del speed-up en $PMC2E_{NF}$.

Para realizar los experimentos cuyos resultados se muestran en las tablas 5.1, 5.4 y 5.5 se han generado M.E.S. de la misma forma que en el capítulo anterior.

La introducción de nodos ficticios se hace rentable, tal y como se mostrará y explicará en las tablas 5.2 y 5.3, cuando existen numerosas ecuaciones con variables similares dentro de un sistema grande. Este hecho se da en la realidad cuando se construye un modelo a partir de muchos modelos de menor tamaño. Por ejemplo,

en el modelado de variables de carácter económico a nivel europeo se construye un M.E.S. a partir de los M.E.S. de cada país añadiendo algunas ecuaciones de unión. Esto hace que el modelo europeo tenga una gran cantidad de variables (todas las de los países) pero, por ejemplo, en las que incorpora el modelo español la mayoría de las ecuaciones utilizarán variables locales. Por ejemplo, en el modelo a escala mundial (gestionado por el proyecto LINK en la universidad de Toronto [8]) incluye al modelo español (gestionado por el CEPREDE [2] o Centro de Predicción Económica y por el Instituto Lawrence R. Klein [3] perteneciente a la Universidad Autónoma de Madrid) más una serie de ecuaciones que relacionan variables del modelo español con otras globales.

Puesto que el fin, para que el simulador genere modelos similares a los descritos en el párrafo anterior, es intentar obtener un sistema grande donde haya unas pocas ecuaciones que tengan cualquier variable y una gran parte de las ecuaciones cuyas variables estén dentro de un subconjunto no muy grande de las del sistema, se generan sistemas aleatorios de la siguiente forma: Se genera un sistema donde las $\frac{N}{4}$ primeras ecuaciones tendrán todas la variables del sistema, y el resto de ecuaciones tendrán solo un número de variables endógenas y exógenas generado entre $3\frac{N}{4}$ y N y $3\frac{K}{4}$ y K, respectivamente. A continuación se procede igual que en la generación de los sistemas descritos anteriormente. Para realizar los experimentos cuyos resultados se muestran en las tablas 5.2, 5.3 y 5.6 se han generado los M.E.S. de esta forma.

La tabla 5.1 muestra una comparativa entre la resolución de un modelo por rotaciones de Givens y añadiendo nodos ficticios. Como se vio en el capítulo anterior, el algoritmo $MC2E_G$ se ve poco afectado en tiempo de ejecución por el aumento de d y se incrementa considerablemente al aumentar N o K. Esta misma propiedad se repite en el algoritmo $MC2E_{NF}$, lo cual es lógico puesto que se basa también en rotaciones de Givens sobre matrices con ceros desde la fila K-ésima hasta la d-ésima. El tiempo de ejecución del algoritmo AMC también es independiente de d, lo cual es razonable puesto que solo depende de la estructura del sistema y no de los datos de las variables. Se puede observar en la misma tabla que el tiempo de ejecución del algoritmo $MC2E_{NF}$ se ve afectado tanto por el aumento de K como por el aumento de N.

El primer ratio compara los tiempos de resolución del sistema por ambos algoritmos (sin considerar el tiempo de la heurística que busca el AMC). Como se observa, el coste de $MC2E_G$ es aproximadamente cuatro veces mayor que el coste de $MC2E_{NF}$. Utilizar en estos casos el algoritmo $MC2E_{NF}$ puede tener cierto interés. Por ejemplo, cuando se tienen sistemas que hay que resolver continuamente debido a cambios en las variables o incorporación de nuevos datos. En estos casos, en que el sistema no varía, el algoritmo AMC solo sería ejecutado la primera vez puesto que el árbol no variará, siendo el resto de veces más rentable el algoritmo $MC2E_{NF}$ que el $MC2E_G$.

Pero salvo en casos como el anterior, lo normal es que haya que incluir en el tiempo de ejecución la búsqueda del AMC y la resolución del sistema, por lo que el segundo ratio es el relevante, dando el algoritmo $MC2E_G$ aproximadamente cuatro

			Tiempos			Ratio de los tiempos	
N	K	d	$MC2E_G$	AMC	$MC2E_{NF}$	$\frac{MC2E_G}{MC2E_{NF}}$	$\frac{MC2E_G}{AMC+MC2E_{NF}}$
400	400	1000	27.87	94.12	7.71	3.61	0.27
400	400	1500	28.03	94.46	7.99	3.51	0.27
400	600	1000	75.82	151.82	19.80	3.83	0.44
400	600	1500	76.40	148.98	18.73	4.08	0.46
800	800	2000	435.11	2204.85	102.39	4.25	0.19
800	800	2500	439.88	2205.01	103.70	4.24	0.19
800	1000	2000	741.75	3130.04	167.41	4.43	0.22
800	1000	2500	736.48	3148.18	170.98	4.31	0.22

Tabla 5.1: Tiempos de ejecución (en segundos) en Rosebud de los algoritmos $MC2E_G$, AMC y $MC2E_{NF}$, cuando se varía el número de variables endógenas (N), el número de variables exógenas (K) y el tamaño de la muestra (d). Se muestran los ratios de resolución del sistema y del total de tiempos de los algoritmos.

veces menos tiempo de ejecución que el $MC2E_{NF}$.

No es necesario tomar varias medidas en la tabla 5.1 puesto que su fin es ver las grandes diferencias entre ambos algoritmos y lo poco eficiente que puede resultar usar nodos ficticios si la heurística se dispara en tiempo.

La tabla 5.2, sin embargo, estudia el comportamiento del uso de nodos ficticios para problemas en los que sale rentable calcular el árbol y resolverlo. En este caso sí se toman promedios y desviaciones típicas de cinco medidas. Se muestran en la tabla el número de nodos aceptados y rechazados, el ratio de operaciones y el tiempo de ejecución del algoritmo AMC y $MC2E_{NF}$.

Como se puede observar en la tabla 5.2, el número de nodos añadidos al árbol por el algoritmo AMC es mucho menor que el número de nodos que han sido rechazados. Por lo tanto, uno de los principales problemas de esta heurística surge de la gran cantidad de nodos que se evalúan para ser después desechados por no disminuir el coste global del árbol. Es de destacar la gran variabilidad que tiene el número de nodos rechazados y la poca variabilidad que tienen los nodos añadidos. Esto hace pensar que para un tamaño dado del problema, se pueden rechazar muchos o pocos nodos pero el número de nodos que se incorporarán al árbol será más o menos el mismo. También es de resaltar la poca influencia en ambos parámetros que tiene el aumento de K y d y la gran influencia que tiene el aumento de N en los nodos rechazados (que no en los aceptados).

El ratio de operaciones muestra el cociente entre el número de operaciones en coma flotante necesarias para resolver el M.E.S. usando el AMC y sin usarlo. Dicho ratio ha sido calculado dividiendo el total de los costes de los arcos del AMC por el total de los costes de los arcos del árbol inicial. Por ejemplo, el ahorro en el número de operaciones (en promedio) en el primer caso es del 24 %. Como se puede

Tam. del problema			Nodos	Nodos	Ratio de	Tiempo
N	K	d	Añadidos	Rechazados	Operaciones	AMC
20	1000	2000	$8.0_{1.2}$	$80.2_{20.3}$	$0.76_{0.08}$	$0.18_{0.01}$
20	2000	3000	$7.2_{1.1}$	$56.0_{25.5}$	$0.73_{0.07}$	$0.63_{0.06}$
20	3000	4000	$9.8_{1.8}$	$53.0_{20.6}$	$0.79_{0.04}$	$1.60_{0.30}$
20	4000	5000	$8.8_{1.5}$	$60.8_{14.3}$	$0.69_{0.11}$	$2.72_{0.40}$
40	1000	2000	$10.4_{2.9}$	$213.6_{58.3}$	$0.68_{0.03}$	$0.59_{0.07}$
40	2000	3000	$11.6_{3.4}$	$210.8_{86.9}$	$0.69_{0.06}$	$2.19_{0.43}$
40	3000	4000	$10.2_{3.3}$	$207.4_{30.2}$	$0.68_{0.09}$	$4.59_{0.31}$
40	4000	5000	$11.0_{5.2}$	$228.8_{49.5}$	$0.68_{0.08}$	$9.07_{2.05}$
80	1000	2000	$7.8_{3.0}$	$593.0_{49.5}$	$0.67_{0.06}$	$2.78_{0.38}$
80	2000	3000	$10.2_{2.9}$	$561.2_{91.5}$	$0.63_{0.07}$	$10.04_{2.66}$
80	3000	4000	$13.4_{3.6}$	$550.2_{203.0}$	$0.63_{0.07}$	$16.21_{3.69}$
80	4000	5000	$14.6_{3.7}$	$670.0_{122.7}$	$0.67_{0.02}$	$34.11_{6.74}$
160	1000	2000	$15.0_{4.3}$	$1865.6_{453.8}$	$0.65_{0.03}$	$14.75_{5.19}$
160	2000	3000	$12.6_{2.6}$	$1949.6_{493.4}$	$0.63_{0.05}$	$38.13_{12.11}$
160	3000	4000	$14.4_{1.7}$	$2171.0_{382.3}$	$0.62_{0.05}$	$80.38_{12.09}$
160	4000	5000	$16.0_{1.6}$	$2336.0_{385.2}$	$0.65_{0.06}$	$136.56_{27.46}$
320	1000	2000	$16.6_{4.4}$	$6757.2_{479.4}$	$0.64_{0.02}$	$162.46_{10.94}$
320	2000	3000	$13.2_{3.6}$	$7133.4_{800.4}$	$0.66_{0.02}$	$420.65_{144.57}$
320	3000	4000	$16.6_{2.1}$	$7479.8_{634.0}$	$0.63_{0.04}$	$556.28_{59.27}$
320	4000	5000	$15.8_{0.8}$	$7123.4_{672.1}$	$0.62_{0.05}$	$842.07_{73.73}$

Tabla 5.2: Nodos añadidos, rechazados y tiempo de ejecución (en segundos) en Rosebud de AMC, cuando se varía el número de variables endógenas (N), el número de variables exógenas (K) y el tamaño de la muestra (d). Se muestra el promedio (número grande) y la desviación típica (subíndice) de 5 medidas para el mismo tamaño del problema.

observar, la tendencia conforme el tamaño del problema aumenta es de reducir el ratio de operaciones, es decir, cuando el tamaño del problema aumenta, la diferencia entre el número de operaciones necesarias para resolver el M.E.S. usando AMC y sin usarlo, es mayor.

Como se había dicho anteriormente, el tiempo de ejecución del algoritmo AMC se ve muy afectado por el valor N y en menor medida por K (por d no se ve afectado). Hay que resaltar la gran variabilidad mostrada por este parámetro, que en algunas ocasiones llega a ser un tercio del valor promediado. Esto indica una gran dependencia de la estructura del sistema a resolver y, por lo tanto, una baja prescisión a la hora de predecir el tiempo de ejecución.

La tabla 5.3 muestra una comparación detallada de los algoritmos $MC2E_G$ y $MC2E_{NF}$ para M.E.S. con pocas ecuaciones y gran número de variables (de forma similar a los tamaños usados en la tabla anterior). La cuarta columna, denominada Parte Común, muestra los tiempos de las líneas 1 y 2 de ambos algoritmos. La siguiente columna muestra los tiempos de resolución de un sistema mediante el uso del algoritmo $MC2E_{NF}$ sin contemplar el tiempo de creación del árbol (mostrado en la tabla anterior). La siguiente columna hace lo mismo pero resolviendo el sistema mediante el algoritmo $MC2E_G$. Las dos últimas muestran los ratios entre los tiempos medidos. El primer ratio es entre los tiempos de resolución del sistema dados en las columnas 5 y 6, y el segundo muestra la relación entre los dos algoritmos completos, es decir se han sumado la parte común a cada uno y en el caso del algoritmo $MC2E_{NF}$ los tiempos de creación del árbol.

Como se observa en la tabla, el ratio de resolución del sistema es siempre favorable al algoritmo $MC2E_{NF}$, creciendo conforme crece el tamaño del problema y llegando en algunos casos a superar el 80 % de beneficio. Sin embargo, como se ha explicado anteriormente, este ratio solo es significativo en el caso en el que se disponga del Árbol de Mínimo Coste ya creado. En el caso normal (que haya que hallar el AMC), el ratio total indica una comparación entre ambos algoritmos. Se puede ver que existe una relación entre el tamaño de N y de K (el valor de d no afecta) para la cual el algoritmo $MC2E_{NF}$ mejora a $MC2E_G$. Por ejemplo, para $N = 20$ se ve que con $K = 1000$ ambos algoritmos tienen tiempos de ejecución similares, pero conforme se aumenta K el ratio va creciendo llegando a ser el algoritmo $MC2E_{NF}$ un 16 % más rentable. Sin embargo, si aumentamos N, por ejemplo para $N = 160$, un valor de $K = 1000$ daría un ratio a favor del algoritmo $MC2E_G$ pues tardaría el 70 % del tiempo que tarda AMC. Para un valor de $K = 2000$ el tiempo de ejecución de ambos es similar y a partir de ahí el algoritmo $MC2E_{NF}$ tiene un tiempo de ejecución menor.

Resaltar varias cosas, primero el pequeño valor del tiempo de la parte común de ambos algoritmos, lo que indica donde radica la mayor parte del coste computacional (en la resolución del sistema). Y resaltar también que, aunque la variabilidad en los tiempos de ejecución es grande en ambos algoritmos, no ocurre así con los ratios, que tienen una variabilidad baja. Esto tiene una fácil explicación, y es que la dependencia del problema es grande a la hora de medir el tiempo de ejecución en

N	K	d	Parte Común	Resol. sis. $MC2E_{NF}$	Resol. sis. $MC2E_G$	Ratio de resol. sis.	Ratio Total
20	1000	2000	$4.64_{0.04}$	$1.20_{0.18}$	$1.45_{0.22}$	$1.22_{0.12}$	$1.01_{0.02}$
20	2000	3000	$27.84_{0.02}$	$9.58_{1.56}$	$12.32_{2.15}$	$1.29_{0.13}$	$1.06_{0.03}$
20	3000	4000	$80.57_{0.03}$	$39.13_{2.85}$	$49.14_{4.73}$	$1.26_{0.06}$	$1.07_{0.02}$
20	4000	5000	$176.93_{0.21}$	$72.50_{20.75}$	$114.95_{17.57}$	$1.65_{0.31}$	$1.16_{0.05}$
40	1000	2000	$4.60_{0.02}$	$2.65_{0.42}$	$3.39_{0.35}$	$1.29_{0.10}$	$1.02_{0.03}$
40	2000	3000	$27.91_{0.16}$	$18.74_{3.45}$	$23.99_{2.87}$	$1.30_{0.12}$	$1.06_{0.04}$
40	3000	4000	$80.72_{0.13}$	$57.66_{15.39}$	$83.49_{11.66}$	$1.49_{0.21}$	$1.15_{0.05}$
40	4000	5000	$177.53_{0.25}$	$142.94_{32.17}$	$229.08_{22.03}$	$1.65_{0.27}$	$1.24_{0.08}$
80	1000	2000	$4.71_{0.04}$	$5.24_{0.72}$	$7.37_{0.63}$	$1.42_{0.14}$	$0.95_{0.03}$
80	2000	3000	$28.05_{0.14}$	$36.34_{4.52}$	$51.93_{4.63}$	$1.43_{0.07}$	$1.08_{0.05}$
80	3000	4000	$81.58_{0.12}$	$119.47_{26.79}$	$189.90_{21.42}$	$1.63_{0.24}$	$1.25_{0.06}$
80	4000	5000	$178.24_{0.13}$	$288.99_{7.47}$	$466.51_{17.56}$	$1.61_{0.07}$	$1.29_{0.03}$
160	1000	2000	$4.88_{0.12}$	$11.89_{1.09}$	$17.09_{1.21}$	$1.44_{0.07}$	$0.71_{0.13}$
160	2000	3000	$28.60_{0.10}$	$74.52_{8.15}$	$110.15_{10.60}$	$1.48_{0.10}$	$0.99_{0.10}$
160	3000	4000	$81.61_{0.07}$	$233.54_{2.83}$	$405.32_{49.14}$	$1.74_{0.20}$	$1.23_{0.15}$
160	4000	5000	$178.88_{0.15}$	$592.51_{45.85}$	$1008.26_{60.27}$	$1.71_{0.18}$	$1.31_{0.10}$
320	1000	2000	$4.96_{0.05}$	$28.14_{1.42}$	$42.87_{3.36}$	$1.52_{0.05}$	$0.24_{0.01}$
320	2000	3000	$29.09_{0.20}$	$172.06_{9.90}$	$250.99_{11.15}$	$1.46_{0.05}$	$0.47_{0.12}$
320	3000	4000	$82.82_{0.11}$	$511.22_{26.35}$	$892.16_{76.46}$	$1.75_{0.15}$	$0.85_{0.09}$
320	4000	5000	$180.43_{0.49}$	$1206.26_{71.12}$	$2242.92_{149.72}$	$1.87_{0.18}$	$1.09_{0.09}$

Tabla 5.3: Tiempos de ejecución (en segundos) en Rosebud y ratios comparativos de los algoritmos 3 y 5, cuando se varía el número de variables endógenas (N), el número de variables exógenas (K) y el tamaño de la muestra (d). Se muestra el promedio (número grande) y la desviación típica (subíndice) de 5 medidas para el mismo tamaño del problema.

ambos algoritmos, y sin embargo ambos se ven afectados de la misma forma, anulándose dicho efecto en la división (si el modelo a resolver necesita mucho tiempo, lo necesitará para ambos y el ratio seguirá siendo el mismo o muy similar).

Para el estudio experimental de los algoritmos paralelos mostrados en el apartado anterior se han utilizando los mismos generadores de Modelos de Ecuaciones Simultáneas que en el estudio secuencial. Se han desarrollado versiones de los algoritmos en paralelo para memoria compartida utilizando OpenMP y llamadas a LAPACK y BLAS (puesto que como ya se ha dicho, están desarrollados en muchos sistemas para poder ser usados en paralelo en memoria compartida). En los experimentos se han utilizado hasta 8 procesadores.

La tabla 5.4 muestra tiempos de ejecución y speed-ups del algoritmo $PMC2E_{NF}$ para los dos tipos de paralelización descritos en la sección anterior. Los tiempos tomados no contemplan el cálculo del AMC, y por lo tanto el tiempo de la llamada al algoritmo AMC no está sumado. La razón, tal y como se comentó en la tabla 5.1, es que para los tamaños del problema mostrados en la tabla, el tiempo de cálculo del AMC es demasiado grande y no hace rentable el uso del algoritmo $PMC2E_{NF}$ (es más rentable usar $PMC2E_G$). Tan solo sería recomendable su uso en el caso en el que se tuviera que resolver numerosas veces el mismo M.E.S. (por actualización de datos, constante aumento de la muestra, etc.) puesto que el cálculo del AMC solo se haría la primera vez, siendo el tiempo de ejecución el resto de veces similar a los tiempos mostrados en la tabla. En dicho caso hipotético se hace rentable el uso del algoritmo $PMC2E_{NF}$ y su paralelización.

Tal y como muestra la tabla 5.4, los speed-ups obtenidos son altos para tamaños del problema grande. Se puede observar que los tiempos de ejecución de los dos tipos de paralelización son muy similares debido a que el árbol utilizado es relativamente grande y los tiempos de espera en el recorrido en profundidad no sean significativos frente al tiempo total.

Sin embargo, si se consideran los tiempos del algoritmo AMC (que pueden ser observados en la tabla 5.1) los speed-ups son cercanos a uno (incluso con 8 procesadores) en ambos tipos de paralelización, tal y como muestra la tabla 5.5. Aun así esto no tiene una gran importancia puesto que, tal y como se demostró en la sección anterior, en caso de tener que construir el AMC para estos tamaños del problema, el algoritmo sin nodos ficticios tiene menor coste computacional, por lo que sería el usado.

Por lo tanto el interés se centra más en ver como paraleliza el algoritmo para tamaños en los cuales es rentable su uso (dados en la tabla 5.2). La tabla 5.6 muestra los tiempos de ejecución y speed-ups para tamaños del problema en los que se hace rentable el uso de nodos ficticios. Se observa que en general, el speed-up aumenta cuando se incrementa el tamaño del problema en ambos tipos de paralelización. También se observa que los speed-ups obtenidos por la paralelización en amplitud son ligeramente mejores que los obtenidos por la paralelización en profundidad.

N	K	d	1 proc.	2 proc.	Sp	4 proc.	Sp	8 proc.	Sp
Recorrido en profundidad									
400	400	1000	7.71	4.01	1.92	2.37	3.25	1.61	4.79
400	400	1500	7.99	4.08	1.96	2.43	3.29	1.67	4.78
400	600	1000	19.80	9.89	2.00	5.26	3.76	3.22	6.15
400	600	1500	18.73	9.74	1.92	5.11	3.67	3.21	5.83
800	800	2000	102.39	52.19	1.96	28.98	3.53	15.60	6.56
800	800	2500	103.70	52.69	1.97	29.30	3.54	15.70	6.61
800	1000	2000	167.41	87.80	1.91	46.16	3.63	22.99	7.28
800	1000	2500	170.98	89.73	1.91	46.64	3.67	22.86	7.48
Recorrido en amplitud									
400	400	1000	7.71	4.13	1.87	2.4	3.21	1.45	5.32
400	600	1000	18.18	9.33	1.95	5.44	3.34	3.33	5.46
800	800	2000	102.39	52.41	1.95	30.27	3.38	17.5	5.85
800	1000	2000	169.51	90.99	1.86	48.15	3.52	24.13	7.02

Tabla 5.4: Tiempos de ejecución (en segundos) en Rosebud y speed-up correspondientes al algoritmo $PMC2E_{NF}$ sin el cálculo del AMC (no se contempla el tiempo del algoritmo AMC) para ambos tipos de recorrido del árbol, cuando se varía el número de variables endógenas (N), el número de variables exógenas (K) y el tamaño de la muestra (d).

Recorrido en profundidad									
N	K	d	1 proc.	2 proc.	Sp	4 proc.	Sp	8 proc.	Sp
400	400	1000	101.83	98.13	1.04	96.49	1.06	95.73	1.06
400	400	1500	102.11	98.20	1.04	96.55	1.06	95.79	1.07
400	600	1000	171.62	161.71	1.06	157.08	1.09	155.04	1.11
400	600	1500	167.71	158.72	1.06	154.09	1.09	152.19	1.10
800	800	2000	2307.24	2257.04	1.02	2233.83	1.03	2220.45	1.04
800	800	2500	2308.55	2257.54	1.02	2234.15	1.03	2220.55	1.04
800	1000	2000	3297.45	3217.84	1.02	3176.20	1.04	3160.03	1.04
800	1000	2500	3319.16	3237.91	1.03	3194.82	1.04	3178.24	1.04
Recorrido en amplitud									
400	400	1000	102.01	98.25	1.04	96.52	1.06	95.57	1.07
400	600	1000	167.14	158.29	1.06	154.40	1.08	152.29	1.10
800	800	2000	2319.04	2257.26	1.02	2235.12	1.03	2222.35	1.04
800	1000	2000	3323.11	3246.59	1.02	3204.75	1.04	3188.13	1.04

Tabla 5.5: Tiempos de ejecución (en segundos) en Rosebud y speed-up comparativos correspondientes al algoritmo $PMC2E_{NF}$ con el cálculo del AMC (se contempla el tiempo de la llamada al algoritmo AMC) para ambos tipos de recorrido del árbol, cuando se varía el número de variables endógenas (N), el número de variables exógenas (K) y el tamaño de la muestra (d).

N	K	d	1 proc.	2 proc.	Sp	4 proc.	Sp	8 proc.	Sp
Recorrido en profunidad									
20	3000	4000	70.45	62.64	1.12	63.27	1.11	52.41	1.34
20	4000	5000	149.69	109.02	1.37	90.70	1.65	55.96	2.67
40	3000	4000	113.68	61.37	1.85	34.52	3.29	26.53	4.28
40	4000	5000	194.44	129.33	1.50	113.83	1.71	80.05	2.43
80	3000	4000	248.08	205.62	1.21	143.16	1.73	143.16	1.73
80	4000	5000	566.09	431.87	1.31	363.29	1.56	228.66	2.48
160	3000	4000	365.97	222.26	1.65	183.24	2.00	122.78	2.98
160	4000	5000	1329.29	779.17	1.71	521.41	2.55	350.01	3.80
320	3000	4000	854.49	554.00	1.54	357.17	2.39	212.85	4.01
320	4000	5000	2571.66	1482.43	1.73	976.61	2.63	599.32	4.29
Recorrido en amplitud									
20	3000	4000	70.46	58.77	1.20	41.63	1.69	39.34	1.79
20	4000	5000	171.90	159.48	1.08	122.51	1.40	121.85	1.41
40	3000	4000	109.79	60.20	1.82	30.51	3.60	24.23	4.53
40	4000	5000	194.44	128.51	1.51	110.10	1.77	77.80	2.50
80	3000	4000	189.90	118.42	1.60	89.15	2.13	49.54	3.83
80	4000	5000	566.09	328.96	1.72	259.97	2.18	189.38	2.99
160	3000	4000	383.88	252.51	1.52	146.99	2.61	100.61	3.82
160	4000	5000	1329.29	750.13	1.77	425.08	3.13	275.89	4.82
320	3000	4000	932.55	539.48	1.73	300.18	3.11	169.83	5.49
320	4000	5000	2510.03	1339.88	1.87	790.25	3.18	474.77	5.29

Tabla 5.6: Tiempos de ejecución (en segundos) en Rosebud y speed-up correspondientes al algoritmo $PMC2E_{NF}$ sin el cálculo del AMC (no se contempla el tiempo de la llamada al algoritmo AMC) para ambos tipos de recorrido del árbol, cuando se varía el número de variables endógenas (N), el número de variables exógenas (K) y el tamaño de la muestra (d) en valores similares a los de la tabla 5.2.

5.6. Conclusiones

Se han desarrollado nuevos algoritmos que, utilizando la descomposición QR y las rotaciones de Givens, mejoran en ciertos casos los tiempos dados por los algoritmos estudiados en el capítulo anterior. Esto se consigue mediante la inclusión de matrices intermedias que reducen el número de rotaciones de Givens necesarias para hallar la descomposición QR en cada ecuación.

Se ha desarrollado un algoritmo para obtener una aproximación del árbol de mínimo coste donde los nodos corresponden a las matrices asociadas a cada ecuación y otras incluidas para reducir el coste. También se ha desarrollado un nuevo algoritmo para el estimador MC2E que, utilizando el árbol hallado, resuelve las ecuaciones del M.E.S. Dicho algoritmo ha sido paralelizado en memoria compartida.

Se han estudiado y comparado los nuevos algoritmos describiendo en qué casos se obtiene una reducción del coste computacional respecto a los estudiados en capítulos anteriores. También se ha estudiado el speed-up de las dos versiones paralelas desarrolladas según el recorrido del árbol. Se concluye que ambas versiones dan resultados muy similares en problemas donde el árbol es grande, siendo ligeramente superior el speed-up de la paralelización con recorrido en amplitud cuando el árbol es relativamente pequeño.

Capítulo 6

Conclusiones y Trabajos Futuros

En este último capítulo resumimos las principales conclusiones que se pueden extraer del trabajo realizado con el fin de tener una visión general de los resultados obtenidos.

6.1. Conclusiones

Como se ha visto en el capítulo de introducción, un Modelo de Ecuaciones Simultáneas es un conjunto de ecuaciones de regresión donde existe influencia simultánea entre variables y ecuaciones. Nacieron en econometría a mediados del siglo pasado y han ido creciendo de forma paralela al desarrollo de las computadoras debido a su altas necesidades computacionales.

Hoy en día, los M.E.S. se han comenzado a utilizar en otras disciplinas y tienen aplicaciones más diversas que aquellas para las que nacieron.

En este trabajo se han propuesto algoritmos de resolución de M.E.S. basados en la descomposición QR para el estimador de información limitada más utilizado, MC2E.

Se han desarrollado dos nuevos esquemas, uno basado solamente en la descomposición QR mediante el método de las reflexiones de Householder y otro que, además de este método, utiliza rotaciones de Givens para retriangularizar la matriz en cada ecuación. Ambos algoritmos han sido comparados dando mejor resultado el que utiliza rotaciones de Givens.

Se ha desarrollado una versión paralela en memoria compartida del algoritmo basado en rotaciones de Givens obteniéndose speed-ups satisfactorios.

En el algoritmo basado en las rotaciones de Givens es posible reducir el número de rotaciones a usar si se comparten columnas entre matrices. Por ejemplo, si las variables que aparecen en una ecuación están incluidas en otra, se podría resolver primero la segunda y después la primera obteniendo su matriz asociada a partir de la matriz de la segunda, y reduciendo así el número de rotaciones de Givens usadas (y por lo tanto el tiempo de ejecución). Sin embargo, este caso no es muy común, por lo que el ahorro no sería grande. Por otro lado, se podrían utilizar matrices intermedias

que, no correspondiendo a ninguna ecuación, redujeran el número de rotaciones de Givens necesarias para obtener la descomposición QR de otras matrices asociadas a ecuaciones.

Se plantea un algoritmo que obtiene un árbol cuyos nodos corresponden a las matrices descritas (llamados ficticios) y a las matrices correspondientes a cada una de las ecuaciones del M.E.S. Para la obtención del árbol se propone una heurística basada en un conjunto de reglas.

A continuación se propone un nuevo algoritmo basado, como el anterior, en rotaciones de Givens, pero que utiliza dicho árbol para obtener la descomposición QR de las matrices asociadas a cada ecuación con un número menor de retriangularizaciones. Este nuevo algoritmo tiene un coste de ejecución menor que el anterior. Sin embargo, el tiempo de obtener dicho árbol puede ser considerable y hacer que el coste total empeore hasta el punto de dejar de ser rentable. Solo en los casos donde se tenga ya construido el árbol porque se resuelva el mismo modelo varias veces (por actualización de la muestra, por ejemplo), o aquellos donde el número de variables es muy grande en comparación con el número de ecuaciones, es más rentable obtener el árbol y resolver el M.E.S. Este último es el caso de muchos modelos macroeconómicos, los cuales están formados por la unión de modelos más pequeños. Al unir muchos modelos pequeños, el número de variables total es muy grande pero al resolver cada uno de ellos se observa que se tienen unas pocas ecuaciones donde se usa solo una parte pequeña de las variables. En este tipo de modelo la utilización del árbol reducirá considerablemente el coste computacional.

Se desarrollan dos versiones en paralelo de este algoritmo. Una recorriendo el árbol en profundidad y otra en amplitud. La primera da mejores resultados que la segunda en árboles grandes, puesto que el tamaño de memoria ocupada (que puede ser un problema debido a la cantidad de matrices intermedias que se pueden llegar a crear) es menor que en otros recorridos. Esto es debido a que se liberan las matrices asociadas a los nodos ficticios conforme se realizan todas las ecuaciones accesibles desde ellos. Sin embargo, es necesario que unos procesos esperen a otros siendo este tiempo despreciable en árboles grandes pero importante en árboles pequeños. El recorrido en amplitud reduce el tiempo de espera de unos procesos a otros pero aumenta el tamaño de memoria ocupada por el algoritmo. Este recorrido es más rentable en árboles pequeños donde el total de memoria no es importante y el tiempo de espera de un proceso a otro con respecto al tiempo total sí lo es.

6.2. Trabajos futuros

A la vista de los resultados obtenidos, se proponen algunas líneas de investigación (en algunas ya se ha empezado a trabajar) relacionadas y complementarias con el trabajo realizado en esta tesis de master, que se consideran interesantes a corto y medio plazo y que se podrían agrupar en cuatro direcciones:

- Resolución de M.E.S. Se han desarrollado versiones de los algoritmos en memo-

ria compartida. Se plantea como línea futura a corto plazo desarrollar también versiones en paralelo en memoria distribuida, también versiones híbridas y en sistemas heterogéneos. Con lo cual se dispondrá de una serie de funciones que son de utilidad para científicos de diversos campos y en distintos sistemas.

- Árbol de Mínimo Coste. Se ha visto en el capítulo 5, que la versión del algoritmo que utiliza el Árbol de Mínimo Coste ahorra en tiempo de ejecución respecto a la que no lo utiliza. Sin embargo, el tiempo de obtención del árbol hace que dicha versión no sea útil en muchos casos salvo que se disponga previamente del árbol construido. Una línea de trabajo futura consiste en la mejora del algoritmo heurístico que obtiene dicho árbol. Como se ha visto en los experimentos del capítulo 5, el número de nodos que se añaden a un árbol es pequeño en comparación con el número de nodos que se rechazan. Por lo tanto, si se consiguieran obtener ciertas reglas o condiciones para reducir el número de nodos que se analizan, se reduciría considerablemente el tiempo de ejecución.

- Modelos de Ecuaciones Simultáneas No Lineales. Actualmente se está avanzando mucho en la teoría de M.E.S. no lineales y, aún siendo una teoría relativamente reciente, ya se ve claramente la gran necesidad computacional que tiene. Como trabajo futuro a largo plazo se plantea el desarrollo y estudio de algoritmos paralalelos para este tipo de modelos de la misma forma que se ha hecho para los M.E.S. lineales.

- Otras técnicas econométricas. También se plantea como trabajo a largo plazo el desarrollo y estudio de algoritmos paralelos para otras técnicas econométricas como, por ejemplo, los modelos VAR, Modelos de Datos Panel, etc.

6.3. Publicaciones

Los resultados de esta tesis han dado lugar a las siguientes publicaciones y presentaciones en congresos:

- Message-passing Two Steps Least Square Algorithms for Simultaneous Equations Models. Jose J. López-Espín and Domingo Giménez. Proceedings International Symposium on Parallel Processing and Applied Mathematics, Lecture Notes in Compute Science, 4967, 127-136, 2007.

- Algoritmos de paso de mensajes para modelos de ecuaciones simultáneas. Jose J. López-Espín and Domingo Giménez. Meeting on Optimization of Parallel Routines and Applications, 10-12 June 2007, Murcia, Spain.

- Two Steps Least Square Algorithms for Simultaneous Equations Models using the QR decomposition. Jose J. López-Espín, Antonio Vidal and Domingo Giménez. Computational Statistics and Data Analysis (sometido).

Bibliografía

[1] Basic linear algebra subprograms. `www.netlib.org/blas`.

[2] Centro de Predicción Económica. `www.ceprede.com`.

[3] Instituto Universitario de Predicción Económica "Lawrece R. Klein". `web.uam.es/otroscentros/klein`.

[4] Jurgen a doornik. Oxmetrics. `www.doornik.com`.

[5] Linear algebra package. `www.netlib.org/lapack`.

[6] Numerical algorithms group. `www.nag.co.uk`.

[7] Parallel basic linear algebra subprograms. `www.netlib.org/scalapack/pblas_qref.html`.

[8] Project LINK Research Centre. `www.chass.utoronto.ca/link`.

[9] Quantitative micro software. `www.eviews.com`.

[10] Statistical package for the social sciences. `www.spss.com`.

[11] Francisco Almeida, Domingo Giménez, José Miguel Mantas, and Antonio M. Vidal. *Introducción a la programación paralela*. Paraninfo Cengage Learning, 2008.

[12] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. J. Dongarra, J. Du Croz, A. Grenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen. *LAPACK User's Guide*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1995.

[13] Rohit Chandra, Ramesh Menon, Leo Dagum, David Kohr, Dror Maydan, and Jeff McDonald. *Parallel Programming in OpenMP*. Morgan Kauffman, 2001.

[14] Rohit Chandra, Ramesh Menon, Leo Dagum, David Kohr, Dror Maydan, and Jeff McDonald. *OpenMP C and C++ Application Program Interface*. OpenMP Architecture Review Board. `http://www.openmp.org/drupal/mp-documents/cspec20.pdf`, 2002.

[15] J. Choi, J. J. Dongarra, S. Ostrouchov, A. Petitet, D. Walker, and R. Clint Whaley. A proposal for a set of parallel basic linear algebra subprograms. Technical Report CS-95-292, Computer Science Dept. University of Tennesee, May. 1995.

[16] A. C. Darnell and J. L. Evans. *The limits of econometrics*. Gower, 1990.

[17] J. J. Dongarra, J. Du Croz, S. Hammarling, and R. J. Hanson. An extended set of fortran basic linear algebra subroutines. *ACM Transactions on Mathematical Software*, 14:1–17, 1988.

[18] R. J. Epstein. *A history of econometrics*. North-Holland, 1987.

[19] W. Feller. *Introducción a la teoría de probabilidades y sus aplicaciones*. Limusa, 1989.

[20] P. Foschi and E. J. Kontoghiorghes. Estimation of VAR Models: Computational aspects. *Computational Economics*, 21(1-2):3–22, 2003.

[21] C. Gatu and E. J. Kontoghiorghes. Parallel algorithms for computing all possible subset regression models using the QR decomposition. *Parallel Computing*, 29(4):505–521, 2003.

[22] G. Golub and C. F. Van Loan. *Matrix Computations*. The John Hopkins University Press, 1996.

[23] A. Gonschorek, I. Lu, J. Halliwill, J. A. Beightol, J. A. Taylor, H. Painter, and D. Eckberg. Influence of respiratory motor neurone activity on human autonomic and haemodynamic rhythms. *Clinical Physiology*, 21(3):323–334, 2001.

[24] W. Greene. *Econometric Analysis*. Prentice Hall, third edition, 1998.

[25] D. Gujarati. *Basic Econometrics*. McGraw Hill, 1995.

[26] M. Harchol-Balter and P. E. Black. Queueing analysis of oblivious packet-routing networks. In *SODA: ACM-SIAM Symposium on Discrete Algorithms*, pages 583–592, 1994.

[27] R. Henry, I. Lu, L. Beightol, and D. Eckberg. Interactions between CO2 Chemoreflexes and Arterial Baroreflexes. *Am. Journal of Physiology*, 274(43):H2177–H2187, 1998.

[28] M. Hofmann, C. Gatu, and E. J. Kontoghiorghes. Efficient algorithms for computing the best subset regression model for large-scale problems. *Computational Statistics and Data Analysis*, 52:16–29, 2007.

[29] E. J. Kontoghiorghes. *Parallel algorithms for linear models: Numerical methods and estimation problems*. Advances in Computational Economics, 2000.

[30] I. Lu. *Applications of structural equation models: Case studies in biomedical and aerospace engineering research*. Joint Statistical Meeting, 2006.

[31] I. Lu and S. P. Jones. An investigation of the effect of cabin pressure altitude and level of SaO2 on passenger comfort - a simultaneous equation model. Technical Report M&CT-TECH-04-019, Boeing Technical Report Series, 2004.

[32] I. Lu, J. Peixoto, and W. A. Taam. Simultaneous equation model for air traffic in the New York area. In *The Seventh Air Transport Research Society World Conference Journal*, 2003.

[33] M. S. Morgan. *The history of econometric ideas*. Cambridge University Press, 1990.

[34] J. M. Perloff. *Microeconomía*. Pearson, 2004.

[35] Michael J. Quinn. *Parallel Programming in C with MPI and OpenMP*. McGraw Hill, 2004.

[36] W. R. Ressler and M. S. Waters. Female earnings and the divorce rate: a simultaneous equation model. *Applied Economics*, 32:1889–1898, 2000.

[37] P. Yanev, P. Foschi, and E. J. Kontoghiorghes. Algorithm for computing the QR decomposition of a set of matrices with common columns. *Algorithmica*, 39:83–93, 2004.