EL COLEGIO DE MÉXICO
CENTRO DE ESTUDIOS ECONÓMICOS
MAESTRÍA EN ECONOMÍA

TRABAJO DE INVESTIGACIÓN PARA OBTENER EL GRADO DE MAESTRO EN ECONOMÍA

"SOLUCIÓN DE MODELOS DE EQUILIBRIO GENERAL USANDO MATHEMATICA"

ELÍAS RAMÍREZ RAMÍREZ

PROMOCIÓN 1995-1997

ASESOR:

ARTURO PÉREZ MENDOZA

JUNIO DE 2001

Para Elisa, Hazael y Jonathan.

Resumen

Se desarrolla un modelo de n sectores con m factores, el cual se resuelve usando MATHEMATICA usando una estrategia y algoritmo de solución basado en el método de Newton-Raphson y una estrategia y algoritmo de solución basado en un proceso de Tâtonnement y por último se comparan los equilibrios usando la variación compensatoria y la variación equivalente.

Palabras clave.

Modelo de equilibrio general aplicado, Calibración, Método de Newton-Raphson, Proceso de Tâtonnement, Variación compensatoria, Variación equivalente.

Indice

Solución de Modelos de Equilibrio General usando MATHEMATICA.

Elías Ramírez Ramírez

1. Introducción.

Desde la teoría de mercados propuesta por León Walras en 1874 (formalizada por los trabajos de Arrow y Debreu, entre otros) en donde se expone la economía como un sistema de ecuaciones simultáneas donde el equilibrio es la solución de dicho sistema, los economistas han tratado de desarrollar a partir de la estructura abstracta del equilibrio general Walrasiano modelos de economías reales. Dichos modelos resultan útiles para evaluar diferentes políticas, mediante el manejo de los parámetros de la demanda y la producción.

Una definición de modelo de equilibrio general que ha recibido consenso entre los teóricos del equilibrio general se refiere a que es un modelo donde todos los mercados se vacían en el equilibrio.[1]

En este trabajo se presenta la metodología para el desarrollo de programas en MATHEMATICA (versión 3.0 o posterior) la cual puede ser utilizada para la resolución numérica de Modelos de Equilibrio General Aplicado (MEGA)[2]. Dentro de esta metodología se aprovechan las bondades de MATHEMATICA para el manejo de cálculos simbólicos, tales como derivadas, integrales y resolución de ecuaciones diferenciales de manera general sin utilizar valores numéricos.

El uso de un programa que maneje el cálculo simbólico permite tener flexibilidad y rapidez en los cálculos, además de que, a diferencia de programas diseñados para la implementación de modelos de equilibrio general, los cuales funcionan como cajas negras, se puede ir verificando cada una de las etapas del diseño, se puede especificar cualquier tipo de función de producción y/o función de utilidad y se puede depurar el programa para lograr un manejo óptimo de memoria.

---

[1] Definición dada por Shoven J. Y Whalley J., (1984).

[2] Un trabajo desarrollado sobre la solución numérica de modelos de equilibrio general es el realizado por Urzúa, C. (1994), el cual presenta un programa escrito en GAUSS llamado "RESUELVE".

En la siguiente sección, se desarrolla el modelo básico, no sin antes exponer las etapas para realizar un análisis aplicado de equilibrio general. En la sección 3 se definen las ecuaciones del modelo, se calibra, se determina el equilibrio original tanto por el método de Newton-Raphson como con el método de Tâtonnement, se determina el equilibrio con un cambio de política económica y se calculan las variaciones compensatoria y equivalente. Finalmente, en los apéndices se incluyen los diagramas de los algoritmos de solución y los programas escritos en el lenguaje de programación de MATHEMATICA.

2. Modelo Básico.

Antes de la presentación del modelo básico, se expondrán las etapas para la realización de un análisis aplicado de equilibrio general.

2.1. Etapas para un modelo de equilibrio general aplicado.[3]

En el análisis aplicado de equilibrio general se involucra el diseño y uso de un modelo de equilibrio general especificado numéricamente para la evaluación de ciertas políticas económicas.

Las etapas para un modelo de equilibrio general aplicado se presentan en el apéndice A.1. en donde a la economía bajo consideración se le considera en equilibrio y se le llamará : equilibrio original. Las etapas de un modelo de equilibrio general son las siguientes:

(a) Especificación del modelo.
(b) Calibración. Cálculo de las variables exógenas y parámetros y cálculo del equilibrio original.
(c) Especificación del cambio de política económica.
(d) Cálculo del nuevo equilibrio.
(e) Comparación entre el nuevo equlibrio y el equilibrio original.

En la etapa de especificación del modelo se establecen las relaciones entre las variables del modelo, es decir, se especifican las ecuaciones que definen la estructura del modelo y las ecuaciones que especifican su equilibrio.

En la práctica, el equilibrio original se construye a partir de los datos de las cuentas nacionales y otras fuentes de información. Algunos datos son tomados como correctos y otros son ajustados para ser consistentes con los datos que generen el equilibrio original.

---

[3] Adaptado de Shoven J. Y Whalley J., (1984).

En la etapa de calibración[4] se encuentra el valor de los parámetros y las variables exógenas del modelo. Para estar en condiciones de calibrar un modelo se requieren los datos de la economía para un año base o un periodo, los cuales deben de estar especificados en una Matriz de Contabilidad Social[5]. Por otro lado, una convención de unidades muy utilizada, adoptada originalmente por Harberger, es escoger un valor unitario para los precios de los bienes y factores[6]. De esta manera, los resultados de la etapa de calibración incluyen los valores de las variables exógenas y parámetros del modelo, como por ejemplo las elasticidades.

En las etapas de especificación del cambio de política económica y el cálculo del equilibrio nuevo, se determina el valor de la variable exógena o parámetro cuyo cambio será evaluado al calcular el nuevo equilibrio.

Por último, en la etapa de comparación de los equilibrios original y nuevo típicamente se utilizan las medidas conocidas como variación equivalente y variación compensatoria hicksiana.

2.2. El problema del consumidor.

El consumidor representativo de la economía, se supone maximiza su función de utilidad:

$$\text{Max } U = U(X_1, X_2, \ldots, X_n) \tag{1}$$

sujeto a la restricción presupuestal:

$$\sum_{i=1}^{n} p_i X_i = I \tag{2}$$

donde:

X_i : Consumo correspondiente al bien i,
p_i : precio del bien i,
I : ingreso del consumidor representativo.

---

[4] Término acuñado por Ahsan Mansur y Whalley (1984). Como la capacidad del modelo para reproducir los datos del año base como una solución del modelo.

[5] Una matriz de contabilidad social proporciona la base de datos a partir de la cual es relativamente fácil calibrar el modelo e incluye información normalmente incluida en las cuentas nacionales presentada de una manera tal que cada hilera y columna de la matriz representa una cuenta de ingreso y de gasto, respectivamente, de los diferentes agentes de la economía, de manera tal que la suma de cada hilera debe corresponder a la suma de cada columna.

[6] Debido a que los datos endógenos provenientes de una matriz de contabilidad social están dados en términos de valor, se deben escoger las unidades para los bienes y factores que permitan obtener las observaciones de cantidades y precios de manera separada, de esta manera una convención originalmente adoptada por Harberger era establecer los precios para los factores y los bienes iguales a la unidad en el equilibrio original. (Shoven, Whalley, 1984)

Se llamará a la ecuación (1) la función de utilidad social y a la ecuación (2) se le denominará la restricción presupuestal social, o ingreso nacional.

La condición de primer orden para este problema de maximización es que la tasa marginal entre el bien i y el j debe coincidir con la tasa económica de sustitución entre el bien i y el j, es decir:

$$\frac{\partial U / \partial X_i}{\partial U / \partial X_j} = \frac{p_i}{p_j} \qquad i \neq j \quad i,j = 1,\ldots\ldots,n \tag{3}$$

2.3. El problema del productor.

Existen n sectores productivos, se supondrá una firma representativa de cada sector que utiliza m factores de producción con los cuales produce un bien específico. El problema de cada firma es la minimización de costos sujeta a un nivel de producción, es decir:

$$\text{Min} \qquad C_i = \sum_{j=1}^{m} w_j L_{ij} \qquad i = 1,\ldots\ldots,n \tag{4}$$

Sujeto a:

$$Y_i = F_i(L_{i1},\ldots\ldots,L_{im}) \qquad i = 1,\ldots\ldots,n \tag{5}$$

donde:

C_i : Función de costos.
Y_i : Cantidad física del i-esimo bien producido.
$L_{i,j}$: Cantidad del factor de producción j distribuída en el sector i.
F_i : Función de producción homogénea de grado 1.[7]

La condición de primer orden de este problema de maximización es que la productividad marginal del j-ésimo factor es igual a su precio, es decir:

$$\partial F_i / \partial L_{ij} = w_j \qquad i = 1,\ldots,n;\ j = 1,\ldots,m \tag{6}$$

Por lo que la tasa técnica de sustitución entre el factor j y el k es igual a la tasa económica de sustitución entre el factor j y el k, es decir:

$$\frac{\partial F_i / \partial L_{ij}}{\partial F_i / \partial L_{ik}} = \frac{w_j}{w_k} \qquad j \neq k \quad j,k = 1,\ldots\ldots,m;\ i = 1,\ldots\ldots,n \tag{7}$$

---

[7] Una función $f:\mathbb{R}^n \longrightarrow \mathbb{R}$ es de grado 1 si $f(tx)=tf(x)$ para todo $t>0$.

Es decir que la tasa técnica de sustitución entre cualquier par de factores es la misma en todos los sectores de la economía[8].

Por último, la cantidad total de cada factor se supone fija:

$$Ltot_j = \sum_{i=1}^{n} L_{ij} \qquad j=1,\ldots,m \qquad (8)$$

donde $Ltot_j$ es la cantidad total del j-ésimo factor.[9]

2.4. Condiciones de equilibrio.

Se supondrá que todos los mercados están en equilibrio, es decir que la cantidad consumida de cada bien es igual a la cantidad producida de ese bien para cada uno de los mercados, es decir:

$$X_i = Y_i \qquad i=1,\ldots,n \qquad (9)$$

Se supondrá que no existen ganancias extraordinarias en cada sector, esto es que la suma del gasto en factores de producción es igual al ingreso por bien producido en cada uno de los sectores de la economía, es decir.

$$p_i Y_i = \sum_{j=1}^{m} w_j L_{ij} \qquad i=1,\ldots,n \qquad (10)$$

Por último, el ingreso nacional será igual al gasto total en factores de producción, es decir:

$$I = \sum_{j=1}^{m} w_j Ltot_j \qquad (11)$$

Las ecuaciones (2), (3), (5), (7), (8),(9), (10) y (11) definen el modelo, se tienen 3n+nm+m+1 ecuaciones y 3n+nm+m+1 incógnitas. Siendo las incógnitas X_i, L_{ij}, p_i,w_i,Y_i e I. Las variables exógenas son: $Ltot_j$ y los parámetros de las funciones de producción y utilidad: A_i, α_i, β_{ij}.

Sin embargo, de acuerdo a la Ley de Walras[10], el sistema está sobreidentificado, solo necesitamos 3n+nm+m ecuaciones por lo que se debe de dar exógenamente el valor de alguna variable endógena, el cuál se convierte en un valor numerario.

---

[8] Condición de producción para el equilibrio competitivo.
[9] Se supone que todos los factores de producción están siendo totalmente empleados.
[10] La ley de Walras establece que: Si de los n mercados que integran el modelo, n-1 están en equilibrio, entonces el mercado restante lo estará.

3. Resolución del modelo.

3.1. Formas funcionales.

Para estar en condiciones de obtener una solución numérica del modelo, se especificarán las funciones de producción y de consumo.

3.1.1. Función de consumo.

La función que se utilizará para el desarrollo de este trabajo es la función de utilidad Cobb-Douglas:

$$U(X_1, X_2, \ldots, X_n) = \prod_{i=1}^{n} X_i^{\alpha_i} \qquad (12)$$

con $\sum_{i=1}^{n} \alpha_i = 1$

donde:
X_i : representa la cantidad del bien i.
α_i : parámetros de los bienes, los cuales son números positivos que representan las preferencias del consumidor[11].

---

[11] La función Cobb-Douglas tiene las siguientes propiedades respecto al consumo:

A. La tasa marginal de sustitución entre parejas de bienes, es decir cuánto debe cambiar el consumo de un bien para mantener el nivel de utilidad sin cambio cuando el consumo del otro bien cambia, depende de sus parámetros α_i y la cantidad de los dos bienes en estudio, y es independiente de la transformación monotónica que se le haya realizado a la función de utilidad, es decir:

$$TMS = -\frac{\partial U / \partial X_i}{\partial U / \partial X_j} = -\frac{\alpha_i X_j}{\alpha_j X_i}$$

B. La fracción de ingreso que paga el consumidor Cobb-Douglas en el bien i respecto al ingreso total es equivalente a la fracción que representa el parámetro del bien i, α_i, respecto a la suma de los parámetros de los bienes considerados en la función de utilidad.

$$\frac{p_i X_i}{I} = \frac{\alpha_i}{\sum_{1}^{n} \alpha_i}$$

3.1.2. Función de producción.

La función de producción que se usará será del tipo Cobb-Douglas con m factores:

$$Y_i = F_i(L_{i1},....,L_{im}) = A_i \prod_{j=1}^{m} L_{ij}^{\beta_{ij}} \qquad i=1,.....n \qquad (13)$$

con $\sum_{j=1}^{m} \beta_{ij} = 1$. $\qquad i=1,.....,n$

donde:

A_i : representa, estrictamente hablando, la escala de producción, es decir cuánto del producto obtendríamos si utilizáramos una unidad de cada insumo.
L_{ij} : representa la cantidad del insumo i en el producto i.
β_{ij} : parámetro que miden la cantidad de producto que responde a los cambios en los insumos[12].

---

[12] La función Cobb-Douglas respecto a la producción tiene las siguientes propiedades:

A. La tasa técnica de sustitución entre parejas de insumos, es decir la medida de cuánto debe ajustarse uno de los insumos para mantener la cantidad de producto sin cambio cuando el otro insumo cambia, depende de sus parámetros β_{ij} y la cantidad de los dos insumos en estudio, es decir:

$$TTS_i = -\frac{\partial F / \partial L_{ij}}{\partial F / \partial L_{ik}} = \frac{\beta_{ij}\, L_{ik}}{\beta_{ik}\, L_{ij}} \quad , \quad j \neq k$$

B. La elasticidad de sustitución, es decir la medida del porcentaje de cambio de la razón de insumos dividido por el porcentaje de cambio de la tasa técnica de sustitución, es unitaria, es decir:

$$\sigma = \frac{\Delta(L_{ik} / L_{ij}) / L_{ik} / L_{ij}}{\Delta TTS_i / TTS_i} = \frac{d \ln(L_{ij} / L_{ik})}{d \ln TTS_i} = 1 \quad , \quad j \neq k$$

C. La elasticidad de escala, es decir la medida del cambio en porcentaje en el producto debido a un incremento de todos los insumos de 1 por ciento, es igual a la suma de los parámetros de los insumos, es decir:

$$Sea\ Y_i = F(x),\ t\ es\ un\ escalar,\ x\ es\ un\ vector\ de\ insumos:$$

$$e_i(x) = \frac{t}{Y_i(t)} \frac{dY_i(t)}{dt} = \sum_{j=1}^{m} \beta_{ij}$$

De lo anterior se desprenden 3 casos:

a. Retornos crecientes a escala:

$$\sum_{j=1}^{m} \beta_{ij} > 1$$

b. Retornos constantes a escala.

$$\sum_{j=1}^{m} \beta_{ij} = 1$$

c. Retornos decrecientes a escala.

$$\sum_{j=1}^{m} \beta_{ij} < 1$$

3.2. Establecimiento de las dimensiones del modelo.

Antes de calcular la solución al modelo básico presentado anteriormente, se deben de establecer las dimensiones del modelo, esto es, de cuántos sectores consta la economía y cuántos factores de producción son utilizados.

De esta manera, utilizando MATHEMATICA se utiliza la función preconstruída "Input", para preguntar el número de sectores y factores, asignando estos valores a las variables **TSectors** (número de sectores de la economía) y **TFactors** (número de factores de producción), con lo que:

```
In[1]:= TSectors = Input["¿Cuántos sectores?"];
        TFactors = Input["¿Cuántos factores?"];
```

3.3. Definición de ecuaciones de los problemas del Consumidor y del Productor.

Para estar en condiciones de encontrar la solución matemática del modelo, primeramente se deben definir las ecuaciones correspondientes al problema del consumidor y del productor, para lo cual se tiene que:

```
In[1]:= U=Product[X[i]^a[i],{i,1,TSectors}];
        restpresupuestal=List[ingreso== Sum[p[i] X[i],{i,1,TSectors}]];
        Do[costos[i]=Sum[w[j] L[i,j],{j,1,TFactors}],{i,1,TSectors}];
        Do[funcionproduccion[i]=A[i]Product[L[i,j]^b[i,j],{j,1,TFactors}],
                                        {i,1,TSectors}];
        produccion=Table[Y[i]==funcionproduccion[i],{i,1,TSectors}];
```

Las cuales corresponden a las ecuaciones (1), (2), (4) y (5) en el problema del consumidor y del productor en el modelo básico.

3.4. Condiciones de primer orden.

Las utilidades marginales y las productividades marginales se obtienen de la siguiente manera:

```
In[1]:= Do[utilmarginal[i]=D[U,X[i]],{i,1,TSectors}];
        Do[prodmarginal[i,j]=D[funcionproduccion[i],L[i,j]],{i,1,TSectors},
                                        {j,1,TFactors}];
```

Para facilitar el manejo de las ecuaciones que deben satisfacerse en el equilibrio, estas se identifican con nombres simbólicos. Así, las ecuaciones que expresan las condiciones que igualan las tasas marginales de sustitución a las tasas económicas de sustitución, se agrupan con el nombre de **tasamargsust**. Y las ecuaciones que igualan las tasas técnicas de sustitución a las tasas económicas de sustitución de los factores, se agrupan con el nombre de

tasatecnicasust. Con lo cual, las condiciones de primer orden para el problema de maximización de la función de utilidad del productor y de minimización de la función de costos del productor se expresan:

```
In[1]:= tasamargsust=Table[p[i]/p[1]==utilmarginal[i]/utilmarginal[1],
                                        {i,2,TSectors}];
        tasatecnicasust= Flatten[Table[w[j]/w[1]==prodmarginal[i,j]
                        /prodmarginal[i,1],{j,2,TFactors},{i,1,TSectors}]];
```

De esta manera ya se han expresado las ecuaciones (3) y (7) del modelo básico.

3.5. Condiciones de equilibrio.

Las condiciones de equilibrio se expresan de la siguiente manera:

```
In[1]:= Do[sumafactorporsectores[j]=Sum[ L[i,j],{i,1,TSectors}],
                        {j,1,TFactors}];
        ecuacionescantfactores=Table[Ltot[j]==sumafactorporsectores[j],
                        {j,1,TFactors}];
        ecuacionesmercequilibrio=Table[X[i]==Y[i], {i,1,TSectors}];
        ecuacionesganextcero=Table[p[i] Y[i]==costos[i], {i,1,TSectors}];
        factores=Sum[w[j] Ltot[j],{j,1,TFactors}];
        ecuacioningresofactores=List[ingreso==factores];
```

Con lo cual se tienen las ecuaciones (8), (9), (10) y (11).

3.6. Calibración.

Una vez definidas las ecuaciones del modelo, el siguiente paso es realizar la calibración, la cual consta de los siguientes pasos (Apéndice A.2.):

(1) Se Introducen valores de las variables endógenas de acuerdo a una matriz de contabilidad social.
(2) Se suponen los precios de los productos y factores iguales a la unidad.
(3) Se calculan los valores de las variables exógenas :

   a. Coeficientes.
   b. Elasticidades.
   c. Cantidades totales de factores.

Con lo cual, para estar en condiciones de calibrar el sistema, es necesario que se introduzcan los valores de las variables endógenas[13] y además se igualen los precios de los productos y factores a la unidad. Usando Mathematica se expresa:

```
In[1]:= (*Se preguntan los valores de las variables endógenas obtenibles de
          la SAM*)
         Do[L[i,j]=Input["Introduzca la cantidad del factor"],
                  {i,1,TSectors},{j,1,TFactors}];
         Do[X[i]=Input["Introduzca la cantidad de producto"],{i,1,TSectors}];
         Do[Y[i]=X[i],{i,1,TSectors}];

         (* Se iguala el valor de los precios de factores y productos a uno *)
         Do[p[i]=1,{i,1,TSectors}];
         Do[w[j]=1,{j,1,TFactors}];
```

Habiéndose introducido el valor de las variables endógenas y para estar en condiciones de calcular el valor de los parámetros, las ecuaciones se agrupan con nombres simbólicos para facilitar su manejo, en este caso se desarrollan dos listas, una lista que contiene las ecuaciones y otra lista que contiene las incógnitas, para después utilizar la función preconstruida **NSolve** para encontrar el valor de las incógnitas[14]. De esta manera se procede a calcular el valor de los parámetros a[i], de la siguiente forma[15]:

```
In[1]:= (*Se iguala la suma de los parámetros a[i] a uno *)
         sumaexponentesdemanda=Sum[a[i],{i,1,TSectors}];
         ecelasticidaddemanda=List[1==sumaexponentesdemanda];

         (*Se une esta ecuación junto con la ecuación de tasamargsust*)
         ecuacionesdemandasust=Join[tasamargsust,ecelasticidaddemanda];

         (*Se genera una lista de incógnitas referente a las variables a[i]*)
         incognitas1=Table[a[i],{i,1,TSectors}];

         (*Se calcula el valor de los parámetros a[i] usando NSolve*)
         alfas=Flatten[NSolve[ecuacionesdemandasust,incognitas1]];
```

---

[13] Los valores de las variables endógenas se obtienen de una matriz de contabilidad social.

[14] La función NSolve permite obtener una solución numérica de un sistema de ecuaciones lineal usando el método de aproximación de la inversa. De esta manera el único requerimiento para poder encontrar la solución única es que la matriz de coeficientes sea no singular, es decir que exista la matriz inversa (Chiang, pag. 106). Con lo cual es posible encontrar cada uno de los parámetros para poder calibrar nuestro modelo.

[15] El procedimiento utilizado para calcular los valores de las variables exógenas se facilita debido a que para hacer esto se utiliza el supuesto de que los precios son iguales a 1(ver nota 6), con lo que obtener el valor de los parámetros significaría prácticamente despejar las variables y encontrar su valor numérico, y de esta manera el utilizar una función para sistemas de ecuaciones lineales como NSolve funciona correctamente. Sin embargo, en los casos en que no se tengan ecuaciones lineales, es posible utilizar el algoritmo de Newton-Raphson y entonces el requerimiento sería que la matriz de Jacobiano no fuera singular.

Ahora se procede a calcular los parámetros b[i,j]:

```
In[1]:= (*Se iguala la suma de los parámetros b[i,j] a uno*)
        Do[sumaexponentesoferta[i]=Sum[b[i,j],{j,1,TFactors}],
                                              {i,1,TSectors}];
        ecelasticidadoferta=Table[1==sumaexponentesoferta[i],{i,1,TFactors}];

        (*Se une esta ecuación junto con la ecuación tasatecnicasust*)
        ecuacionesauxiliares=Join[ tasatecnicasust,ecelasticidadoferta];

        (*Se genera una lista de incógnitas referente a las variables b[i,j]*)
        incognitas2=Flatten[Table[b[i,j],{i,1,TSectors},{j,1,TFactors}]];

        (*Se calcula el valor de los parámetros b[i,j] usando NSolve*)
        betas=Flatten[NSolve[ecuacionesauxiliares,incognitas2]];
```

Habiéndose calculado el valor de los parámetros b[i,j], se calcula el valor de los coeficientes A[i] de la siguiente manera:

```
In[1]:= (*Se genera una lista de incógnitas referente a las variables A[i]*)
        incognitas3=Table[A[i],{i,1,TSectors}];

        (*Se calcula el valor de los coeficiente A[i] usando NSolve*)
        coeficientesA=Flatten[NSolve[produccion,incognitas3]] /.betas;
```

Después se calcula el valor de los parámetros Ltot[j] :

```
In[1]:= (*Se genera una lista de incógnitas referente a las variables L[j]*)
        incognitas4=Table[Ltot[j],{j,1,TFactors}];

        (*Se calcula el valor de los parámetros Ltot[j] usando NSolve*)
        totalfactores=Flatten[NSolve[ecuacionescantfactores,incognitas4]];
```

Por último, los parámetros calculados se agrupan en una variable denominada **parametros**, la cual posteriormente será utilizada para realizar los cálculos del equilibrio original y nuevos equilibrios.

```
In[1]:= parametros=Join[coeficientesA,totalfactores,alfas,betas];
```

Como ejemplo, supóngase una economía que tiene una matriz de contabilidad social con los siguientes datos[16]:

	Sector 1	Sector 2	Consumo	Salarios	Ganancias
Sector 1			20		
Sector 2			20		
Consumo				25	15
Salarios	15	10			
Ganancias	5	10			

De esta manera los valores que se introducen al programa son los siguientes:

TSectors=2
TFactors=2

[16] Ejemplo tomado de Pérez, A. Y González, A. (2000).

L[1,1]=15
L[1,2]=5
L[2,1]=10
L[2,2]=10
X[1]=20
X[2]=20

Como resultado, los valores de los parámetros y las variables exógenas son los siguientes:

```
In[1]:= Print[parametros];
Out[1]:= {A[1]→1.75477,A[2]→2.,Ltot[1]→25.,Ltot[2]→15.,a[2]→0.5,a[1]→0.5,
          b[1,2]→0.25,b[2,2]→0.5,b[1,1]→0.75,b[2,1]→0.5}
```

3.7. Cálculo del equilibrio original.

Habiéndose establecido usando nombres simbólicos las ecuaciones a satisfacer en el equilibrio, se está en condiciones de calcular el equilibrio original, es decir los precios y cantidades. Para lo cual se propone una estrategia de solución[17] y un algoritmo de solución basado en el método de Newton-Raphson (Apéndice E) y una estrategia de solución basada en el mercado de factores junto con un algoritmo de solución basado en un proceso de Tâtonnement.

3.7.1. Estrategia y algoritmo de solución basados en el Método de Newton-Raphson.

Primeramente debe recordarse que ley de Walras establece que el equilibrio de n-1 mercados implica el equilibrio en el n-ésimo mercado, lo cual significa que para poder utilizar el modelo de Newton-Raphson y poder encontrar la matriz inversa del Jacobiano, se hace necesario eliminar una ecuación del sistema básico, ya que de no hacerlo, la matriz del Jacobiano sería una matriz singular y no se podría aplicar el método de N-R.

De esta manera en la estrategia que se utilizará para encontrar la solución basado en el método del algoritmo de Newton-Raphson (Apéndice A.3.) se establecen los siguientes pasos:

(a) Se agrupan las (n-1) ecuaciones.
(b) Se le dan valores iniciales a las variables.
(c) Se agrupan las variables incógnita con sus respectivos valores iniciales y sus límites inferior y superior.

---

[17] El propósito de una estrategia de solución es, de acuerdo a Dervis, De Melo y Robinson (1984), apéndice B, establecer numéricamente un conjunto de funciones no lineales simultáneas cuya solución proporcionará los valores de equilibrio de las variables endógenas del modelo. Mientras que un algoritmo de solución es una técnica computacional para resolver numéricamente el conjunto de ecuaciones no lineales simultáneas.

(d) Se utiliza una función de MATHEMATICA basada en el método de Newton-Raphson a fin de encontrar la solución (valores de equilibrio).

Tomando en consideración lo establecido respecto a la ley de Walras, no se considerará la última ecuación de las ecuaciones **ecuacionescantfactores,** la cual incluía las ecuaciones de equilibrio en los mercados de factores, renombrándose con el nombre de **ecuacionescantfactaux**. Usando MATHEMATICA, lo anterior se realiza de la siguiente manera:

```
In[1]:= ecuacionescantfactaux=Drop[ecuacionescantfactores,-1];
```

Las demás ecuaciones del modelo básico correspondientes a las tasas marginales de sustitución (3) y las restricciones presupuestales (2) se agrupan en la lista de ecuaciones denominada **ecuacionesdemanda**, y las ecuaciones correspondientes a las tasas técnicas de sustitución (7) y las funciones de producción (5) se agrupan en la lista de ecuaciones denominada **ecuacionesoferta**.

```
In[1]:=  (*Une las ecuaciones para formar las ecuaciones de demanda *)
         ecuacionesdemanda=Join[tasamargsust,restpresupuestal];

         (*Une las ecuaciones para formar las ecuaciones de produccion *)
         ecuacionesoferta=Join[tasatecnicasust,produccion];
```

Como siguiente paso, se agrupan todas las ecuaciones correspondientes al modelo básico: las ecuaciones de demanda (2 y 3), las ecuaciones de oferta (5 y 7), las ecuaciones de cantidad total de cada factor (8), las ecuaciones de exceso de demanda en el mercado de bienes igual a cero (9), las ecuaciones de ganancias extraordinarias igual a cero (10) y la ecuación de ingreso nacional (11), conocidas como **ecuacionesdemanda**, **ecuacionesoferta**, **ecuacionescantfactaux**, **ecuacionesmercequilibrio**, **ecuacionesganextcero** y **ecuacionesingresofactores** respectivamente, en una lista de ecuaciones llamada **ecuacionesincognitas** apropiada para aplicar el método de Newton-Raphson:

```
In[1]:=  ecuacionesincognitas=Join[ecuacionesdemanda,
         ecuacioningresofactores,ecuacionesoferta,ecuacionesganextcero,
         ecuacionesmercequilibrio,ecuacionescantfactaux] /. parametros;
```

Obsérvese en la expresión anterior el uso de la regla de transformación /. en donde se sustituyen los valores de **parametros** en las ecuaciones que contiene **ecuacionesincognitas.**

Debido a que la función preconstruida **FindRoot** (basada en el método de Newton-Raphson) falla cuando no se puede calcular la derivada simbólica o cuando es muy lenta la convergencia al resultado, entonces se usará una función escrita por Silvio Levy denominada **MyFindRoot.m** que calcula las derivadas por aproximaciones numéricas, la cual se transcribe en el apéndice D.

Para este fin se carga el programa con la siguiente instrucción:

```
In[1]:=<<Mega`MFRoot`
```

Debido a que **MyFindRoot** requiere valores iniciales para las variables, los valores para las variables correspondientes a los factores de producción L[i,j] se establecen, siguiendo a Noguchi (1991), cercanos al máximo valor para cada variable, esto es:

```
In[1]:=  prec=1. 10^-6;
         Do[Linicial[i,j]=Ltot[j]-prec,{j,1,TFactors},{i,1,TSectors}];
```

Y los valores iniciales para las variables restantes se establecen en la unidad.

Se le da formato a las variables incógnitas.

```
In[1]:=  (* Se le da formato a las variables con sus variables iniciales *)
         (* para que puedan ser procesados *)

         Do[varL[i,j]=List[List[L[i,j],Linicial[i,j],0,Ltot[j]]],
         {j,1,TFactors},{i,1,TSectors}];
         Do[varCons[i]=List[List[X[i],1,0,100000]],{i,1,TSectors}];
         Do[varProd[i]=List[List[Y[i],1,0,100000]],{i,1,TSectors}];
         Do[varPrecio[i]=List[List[p[i],1,0,100000]],{i,1,TSectors}];
         Do[varwages[j]=List[List[w[j],1,0,100000]],{j,1,TFactors}];
        varIngreso=Flatten[List[List[ingreso,1,0,100000]]];
```

Se establece como numerario al precio del factor de producción 1, esto es:

```
In[1]:=  w[1]=1;
```

Por último se elimina dentro de las incógnitas a la variable numerario y se agrupan las variables en una lista que contiene a las variables incógnitas denominada **variablesincognitas.**

```
In[1]:=  (*Se agrupan las variables por variable tipo*)
         variables1=Flatten[Table[varL[i,j],{i,1,TSectors},{j,1,TFactors}]];
         variables2=Flatten[Table[varCons[i],{i,1,TSectors}]];
         variables3=Flatten[Table[varProd[i],{i,1,TSectors}]];
         variables4=Flatten[Table[varPrecio[i],{i,1,TSectors}]];
         variables5=Flatten[Table[varwages[j],{j,1,TFactors}]];

         (* Se elimina de las incógnitas al numerario, que en este caso
               es w[1] *)
         variables5bis=Drop[variables5,{1,4}];

         (* Se unen todas las variables incognitas en la variable variablesincaux *)
         variablesincaux= Join[variables1,variables2,variables3,
               variables4,variables5bis,varIngreso]/. parametros;

         (*Se le da formato a la lista de variables para poder utilizarla en
               MyFindRoot, separando a las variables en subconjuntos de cuatro
               elementos {nombre de la variable, valor inicial, valor mínimo
               ,valor máximo} *)
         variablesincognitas=Partition[variablesincaux,4];
```

Ya habiéndose agrupado las ecuaciones y las incógnitas, se está en condiciones de calcular el equilibrio original usando la función **MyFindRoot**, para lo cual se crea una función denominada **equilibrio**, con la cual se calcula el equilibrio original.

```
In[1]:= (* Ahora se define una función para el cálculo del equilibrio *)
        equilibrio[datos_,sotad_]:=MyFindRoot @@ (Prepend[datos,sotad]);

        (* Y se procede a calcular el equilibrio original *)
        equilibriooriginal=equilibrio[variablesincognitas,
                                      ecuacionesincognitas];
```

Y continuando con el ejemplo anterior :

```
In[2]:= Print["\n El equilibrio original es: \n",equilibriooriginal];

Out[2]:= El equilibrio original es:
         {L[1,1]→15.,L[1,2]→5.,L[2,1]→10.,L[2,2]→10.,X[1]→20.,X[2]→20.,
         Y[1]→20.,Y[2]→20.,p[1]→1.,p[2]→1.,w[2]→1.,ingreso→40.}
```

3.7.2. Estrategia de solución basada en el mercado de factores y algoritmo de solución basado en un proceso de Tâtonnement.

La estrategia de solución que se utilizará en este trabajo es una basada en el mercado de factores (Apéndice A.4) en donde de acuerdo al modelo utilizado, la estrategia es la siguiente:

(a) Se proponen los precios de los factores.

```
In[1]:= (* El valor inicial que se propone es w[j]=1 para j=1,....,TFactors *)
        Do[w[j]=1,{j,1,TFactors}];
```

(b) Se calcula el nivel de ingreso del consumidor representativo (ecuación 11).

```
In[1]:= solingreso=Flatten[Solve[ecuacioningresofactores,{ingreso}]]
                                                        /.parametros;
```

(c) Se calculan las demandas condicionadas de factores por unidad de producto (usando las ecuaciones 5,7 y 13)[18].

---

[18] Usando las ecuaciones 5, 7 y 13 se obtienen las fórmulas de las demandas condicionadas de factores, es decir:

$$L_{ij} = \begin{pmatrix} Y_i \\ A_i \end{pmatrix} \left(\frac{\prod_{k=1, k\neq j}^{m} w_k^{\beta_{ik}}}{w_j^{1-\beta_{ij}}} \right) \left(\frac{\beta_{ij}^{1-\beta_{ij}}}{\prod_{k=1, k\neq j}^{m} \beta_{ik}^{\beta_{ik}}} \right) \quad \text{para } j=1,\ldots,m;\ i=1,\ldots,n$$

```
In[1]:=  (* Se insertan los valores de los parámetros en las ecuacionesoferta y
         se le da el nombre de ecuacionesofertaconparametros *)

         ecuacionesofertaconparametros=ecuacionesoferta/.parametros;


         (* Se crea una lista de las incógnitas *)

         listademandasporproducto= Flatten[Table[L[i,j],{i,1,TSectors},
                                                    {j,1,TFactors}]];

         (* Obtiene los valores de la demanda condicionada por producto*)

         demandascondporproducto= Flatten[Solve[ecuacionesofertaconparametros,
                                    listademandasporproducto]]/.solucion;
```

(d) Habiendo calculado demandas de factores por unidad de producto y teniendo los precios de los factores, los precios de los bienes pueden ser obtenidos utilizando la ecuación (10).

```
In[1]:=  (* Genera una lista de las ecuaciones con los valores calculados*)

         ecuacionesprecios=ecuacionesganextcero/.demandascondporproducto
                                                    /.solucion;

         (*Genera una lista de las incógnitas*)

         listaprecios=Flatten[Table[p[i],{i,1,TSectors}]];

         (*Obtiene los valores de los precios*)

         precios=Flatten[Solve[ecuacionesprecios,listaprecios]];
```

(e) Con los precios de los bienes y el ingreso se puede calcular la demanda de cada uno de los bienes (ecuaciones 2 ,3 y 12)[19]

```
In[1]:=  (* Se genera una lista de las incógnitas de las demandas por bienes *)

         listademandasbienes=Flatten[Table[X[i],{i,1,TSectors}]];

         (*Obtiene el valor de las demandas por bienes *)

         demandas=Flatten[Solve[ecuacionesdemanda,listademandasbienes]]
                                    /.parámetros/.solingreso/.precios;
```

(f) Utilizando la ecuación 9 y suponiendo equilibrio en el mercado de bienes, se obtiene el valor de la oferta de cada bien.

```
In[1]:=  ofertas=Flatten[Solve[ecuacionesmercequilibrio,listaofertabienes]]
                                                    /.demandas;
```

---

[19] Usando las ecuaciones 2, 3 y 12 se obtienen las fórmulas de las demandas por bien, es decir:

$$X_i = \frac{\alpha_i I}{p_i} \qquad \text{para } i = 1,\ldots,n$$

(g) Con los valores obtenidos para la oferta de cada bien se pueden calcular los valores de las demandas condicionadas de cada factor (multiplicando los valores obtenidos en el inciso c) por el valor de la producción de cada bien.

```
In[1]:=  (* Se generan las ecuaciones donde se igualan las demandas de los
         factores al producto de las demandas de factores por producto
          multiplicadas por la cantidad de bienes producidos*)

         ecuacionesdemandafactores=Flatten[Table[demfactor[i,j]==L[i,j]*Y[i],
         {i,1,TSectors},{j,1,TFactors}]];

         (* Se genera la lista de las incógnitas de demanda de factores *)

         listademandafactores=Flatten[Table[demfactor[i,j],{i,1,TSectors},
                                                          {j,1,TFactors}]];

         (* Se obtiene el valor de las demandas de factores *)

         demandafactores=  Flatten[Solve[ecuacionesdemandafactores,
               listademandafactores]]/.ofertas/.demandascondporproducto;
```

(h) Se calcula la demanda total de los factores usando la ecuación 8:

```
In[1]:=  Do[Totaldemandafactor[j]=Sum[demfactor[i,j],{i,1,TSectors}],
                                                          {j,1,TSectors}];
```

(i) Se verifica si existe equilibrio en el mercado de factores, es decir, se determina si el excedente de demanda de los factores es igual a cero.

```
In[1]:=  Excesos=Table[
         excesodemanda[j]=Totaldemandafactor[j]-Ltot[j],{j,1,TFactors}]
                                        /.demandafactores/.parametros;
```

(j) Si no existe equilibrio en el mercado de factores se hace necesario proponer nuevos precios para el mercado de factores e iniciar el proceso a partir del inciso b). Es a esta última parte a la que se le denominará el algoritmo de solución y consiste en que si se verifica que no se tiene equilibrio en el mercado de factores, entonces se debe proponer un nuevo precio para los factores. De esta manera la función que se propone para el nuevo precio es la siguiente[20]:

---

[20] Una consecuencia de que el modelo cumpla con la ley de Walras, es que las trayectorias seguidas por los precios durante el ajuste no exploten y se restrinjan a una esfera k-dimensional, donde k es el número de precios en el mercado de factores. Sin embargo el estar restringido a una esfera k-dimensional no garantiza la convergencia de nuestro sistema, con lo cual la única condición que garantiza la convergencia al precio de equilibrio es que las funciones de demanda obedezcan el Axioma Débil de la Preferencia Revelada (si p^* es un vector de precios de equilibrio, entonces $p^*z(p)>0$ para todo $p\neq p^*$, con $z(p)$ el vector de funciones de exceso de demanda). (Mas-Collel, et al. Pag. 623). O visto de una manera matemática el equilibrio es estable cuando los eigenvalores de la matriz Jacobiana de los excesos de demanda son negativos o tienen una parte real negativa. (Simon,C Y Blume L, Pag.687).

$$w_j(t) = w_j(t-1) + \theta(EDF_j) \qquad j=1,\ldots,m \qquad (14)$$

donde:

$w_j(t)$: Precio del factor en la iteración t.
$w_j(t\text{-}1)$: Precio del factor en la iteración t-1.
θ: Escalar que determina el tamaño del siguiente paso.
EDF_j: Exceso de demanda del factor j calculado en la iteración t-1.

De esta manera, si el exceso de demanda del factor es positiva, entonces el precio del factor se incrementará por un valor igual al producto del escalar por el exceso de demanda del factor.

```
In[1]:=  (* Se establece el parámetro de precisión *)

         prec = 0.001

         (* Se establece el parámetro de ajuste
               w(t+1) =w(t)+[Theta](Exceso de demanda)*)

         Theta = 0.01;

         (*Se verifican los excesos de demanda y entonces se ajustan
                        los precios de los factores *)

         For[j = 1,  j <= TFactors, j++,
               If[Excesos[[j]] <= prec , Continue[ ],
                        (w[j] += Theta*Excesos[[j]];Break[ ])]];
```

Respecto al ejemplo utilizado anteriormente, cuando se llega a excesos de demanda en los factores iguales a cero, los valores de las variables endógenas calculadas usando una estrategia Tâtonnement corresponden a los calculados usando el método de Newton-Raphson. Por último en el apéndice C se presenta el programa en MATHEMATICA correspondiente al método de Tâtonnement.

3.8. Especificación del cambio de política económica, cálculo del nuevo equilibrio y su comparación con el equilibrio original.

3.8.1. Especificación del cambio de política económica y cálculo del nuevo equilibrio.

Una vez establecido el valor de las variables exógenas y habiendo comprobado que se obtienen los valores del equilibrio original, se está en condiciones de probar el modelo con respecto a un cambio en las variables exógenas (o un cambio en la política económica). Para lograr lo anterior se supondrá que la cantidad total del factor de producción j es decir $Ltot_j$ cambiará

de valor[21]. De esta manera, lo primero que se debe de hacer es preguntar el factor a modificar y actualizar la cantidad total de ese factor.

```
In[1]:=  numerofactor=Input["¿Qué factor desea modificar?"];
         nuevovalor=Input["Introduzca el nuevo valor"];
```

Con el nuevo valor, se actualiza la lista denominada **parametros**, la cual se denominará **parametrosnuevos** .

```
In[1]:=  parametrosnuevos=ReplacePart[parametros,
                    Ltot[numerofactor]->nuevovalor,2+numerofactor];
```

Se actualizan las listas **ecuacionesincognitas** y **variablesincognitas**, generándose las nuevas listas **ecuacionesincognitasnuevas** y **variablesincognitasnuevas**, respectivamente.

```
In[1]:=  variablesincauxnuevas=Join[variables1,variables2,variables3,
              variables4,variables5bis,varIngreso]/. parametrosnuevos;

         variablesincognitasnuevas=Partition[variablesincauxnuevas,4];

         ecuacionesincognitasnuevas=Join[ecuacionesdemanda,
              ecuacioningresofactores,ecuacionesoferta,
              ecuacionesganextcero,ecuacionesmercequilibrio,
              ecuacionescantfactaux] /.parametrosnuevos;
```

De esta manera se obtiene el nuevo equilibrio usando la función **equilibrio[datos_,sotad_]** definida anteriormente:

```
In[1]:=  equilibrionuevo= equilibrio[variablesincognitasnuevas,
                                   ecuacionesincognitasnuevas] ;
```

Considerando el ejemplo anteriormente mencionado, si se tuviera un cambio en la cantidad total del factor 2 y pasara de 15 a 19.5 (un incremento del 30%), los nuevos parámetros serían :

```
In[1]:= Print[parametrosnuevos];

Out[1]:= {A[1]→1.75477,A[2]→2.,Ltot[1]→38.,Ltot[2]→19.5,a[2]→0.5,a[1]→0.5,
          b[1,2]→0.25,b[2,2]→0.5,b[1,1]→0.75,b[2,1]→0.5}
```

Y el nuevo equilibrio es:

```
In[2]:=  Print["\n El equilibrio nuevo es: \n",equilibrionuevo];

Out[2]:=  El equilibrio nuevo es:
          {L[1,1]→15.,L[1,2]→6.5,L[2,1]→10.,L[2,2]→13.,X[1]→21.3558,
          X[2]→22.8035,Y[1]→21.3558,Y[2]→22.8035,p[1]→0.936514,
          p[2]→0.877058,w[2]→0.769231,ingreso→40.0}
```

---

[21] Para entender lo anterior supóngase que el factor *j* que se está considerando fuera capital, de esta manera lo que pudiese provocar un incremento en el stock de capital podría ser un incremento en inversión extranjera directa por ejemplo debido a un relajamiento de las leyes que regulen la inversión extranjera directa, de manera tal que esta inversión se distribuiría entre todos los sectores de acuerdo a la demanda por el capital y los precios de equilibrio, materializándose en bienes de capital listos para producir. El nuevo equilibrio representaría el efecto de este incremento en las variables endógenas de la economía.

3.8.2. Comparación entre el equilibrio original y el equilibrio nuevo.

Una pregunta que surge a partir del cambio en la política económica, es si este cambio significa como consecuencia una mejora (o empeoramiento) en el bienestar de los consumidores. Para determinar lo anterior, las medidas que comúnmente se emplean son la variación compensatoria y la variación equivalente. La variación compensatoria (VC) usa los precios y el ingreso del nuevo equilibrio y pregunta cuánto dinero habría que darle al consumidor si se le quisiera compensar por el cambio en la política económica. La variación equivalente (VE) pregunta cuánto dinero habría que quitarle al consumidor antes de instrumentarse el cambio en la política económica, para que disfrutara del mismo bienestar después del cambio[22].

Considerando lo anterior, para una mejora en el bienestar, la VC debería ser negativa y la VE debería ser positiva. Para efectos de este trabajo se puede tomar una convención de signo en la cual un valor positivo indicaría una mejora en el bienestar.

Debido a que en la formulación del modelo se uso una función de utilidad homogénea de grado uno (ecuación 12), adoptando la convención de signo en que una mejora sea representada por una cantidad positiva, las medidas de bienestar se calculan de la siguiente manera:

$$VC = \frac{\left(U^N - U^O\right) I^N}{U^N} \tag{13}$$

$$VE = \frac{\left(U^O - U^N\right) I^O}{U^O} \tag{14}$$

donde:

U^N: Utilidad calculada con los valores del equilibrio nuevo.
U^O: Utilidad calculada con los valores del equilibrio original.
I^N: Ingreso calculado con los valores del equilibrio nuevo.
I^O: Ingreso calculado con los valores del equilibrio original.

Habiendo calculado los valores para el equilibrio original y para el equilibrio nuevo, con MATHEMATICA las variaciones compensatoria y equivalente se obtienen de la siguiente manera:

```
In[1]:= (* Se calculan las utilidades original y nueva y los ingresos
           original y nuevo *)

         Utilidadoriginal=U/.equilibriooriginal/.parametros;
         Utilidadnueva=U/.equilibrionuevo/.parametrosnuevos;
```

[22] Algunas ventajas de usar las medidas de variación compensada y equivalente sobre el cálculo directo de la utilidad son que el valor de la utilidad es un número sin unidades y usando la variación compensada y equivalente se conoce la magnitud del cambio del bienestar en términos monetarios y se puede comparar el costo o el beneficio que tiene alguna política especifícia para el consumidor representativo.

```
ingresooriginal=consumobienes/.equilibriooriginal;
ingresonuevo=consumobienes/.equilibrionuevo;

(*Se calcula la variación compensada*)

variacioncompensada=(Utilidadnueva-Utilidadoriginal)*
 ingresonuevo/Utilidadnueva;

(*Se calcula la variación equivalente*)

variacionequivalente=(Utilidadnueva-Utilidadoriginal)*
 ingresooriginal/Utilidadoriginal;
```

Con respecto al ejemplo, los valores de la variación compensada y equivalente son:

```
In[2]:=  Print["\n La variación compensada es: \n",variacioncompensada];
         Print["\n La variación equivalente es: \n",variacionequivalente];

Out[2]:=  La variación compensada es: 3.74806
          La variación equivalente es: 4.13557
```

4. Conclusiones.

En este trabajo se presentó una metodología completa para obtener una solución numérica de un modelo de equilibrio general simple usando un programa escrito en el lenguaje de programación de MATHEMATICA (versión 3.0). Primeramente se definió el modelo, se obtuvieron los valores de las variables exógenas (calibración) y para la resolución del modelo (cálculo de las variables endógenas en el equilibrio) se establecieron dos estrategias y algoritmos de solución diferentes para obtener los valores del equilibrio: la estrategia y algoritmo de solución basados en el método de Newton-Raphson y la estrategia y algoritmo de solución basado en un proceso de Tâtonnement. Dentro de las ventajas de usar MATHEMATICA para resolver simbólicamente el modelo se tiene que no es necesario, explícitamente definir las funciones de demanda de los bienes ni las funciones de demanda condicionada de los factores, además de que durante la elaboración del programa es posible ejecutar las instrucciones de forma independiente, manteniendo en memoria los valores de las variables hasta que se cierre el programa, y de esta manera realizar las depuraciones con mucha rapidez, con lo que no es necesario compilar todo el programa antes de correrlo.

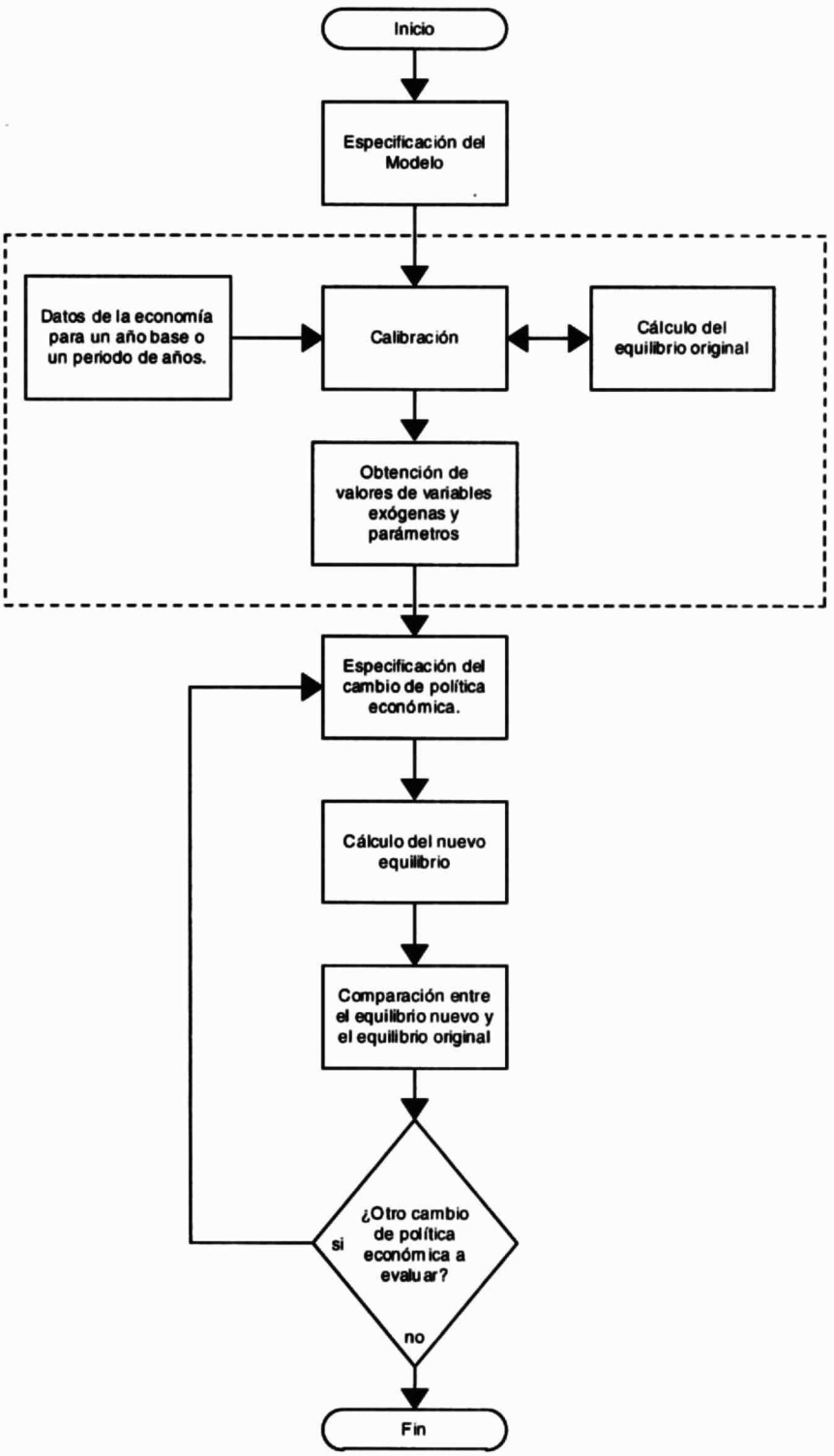
Respecto al método de Newton-Raphson lo que se pide es que las funciones de producción y de consumo sean C^1 y que el Jacobiano de las ecuaciones de equilibrio sea invertible (aunque esto se puede evitar eliminando la ecuación que sea una combinación lineal de las demás). Lo cual significa que el método funciona bien para calcular el equilibrio con cualquier función de producción y de utilidad cuya característica sea que se pueda derivar una sola vez. Esto representaría una ventaja sobre otros programas que no incorporan la posibilidad del cálculo simbólico (tales como GAUSS) y entonces requieren que el usuario defina o mejor dicho obtenga previamente las funciones de demanda de los bienes y las funciones de demanda de los factores antes de poder utilizar el programa, además de que en programas diseñados para la

implementación de modelos de equlibrio general (tales como HERCULES) su funcionamiento es como caja negra en donde las funciones de producción y consumo se encuentran limitadas a ciertas funciones estándar (entre las cuales se tiene la CES o la Cobb-Douglas), con lo que se pierde la flexibilidad de manejar funciones diferentes a las predefinidas por el programa y la posibilidad de obtener o modificar otros resultados que pueden ser útiles para el usuario.

Respecto a la metodología basada en la estrategia y algoritmos de solución de Tâtonnement, aunque incorpora la ventaja de que no es necesario que el Jacobiano de las ecuaciones sea invertible, se deben conocer previamente las ecuaciones de las demandas condicionadas de factores y las demandas de los bienes, lo cuál en este trabajo se evitó debido a que se conocían las propiedades de las funciones de producción y de utilidad. Además de que si existen interacciones entre los mercados este método se vuelve difícil de implementar o ineficiente. De lo anterior se deduce la necesidad de que el programador conozca las propiedades del modelo para poder aplicar la metodología de Tâtonnement; pero una vez realizado el programa para el modelo particular, el método funciona eficientemente.

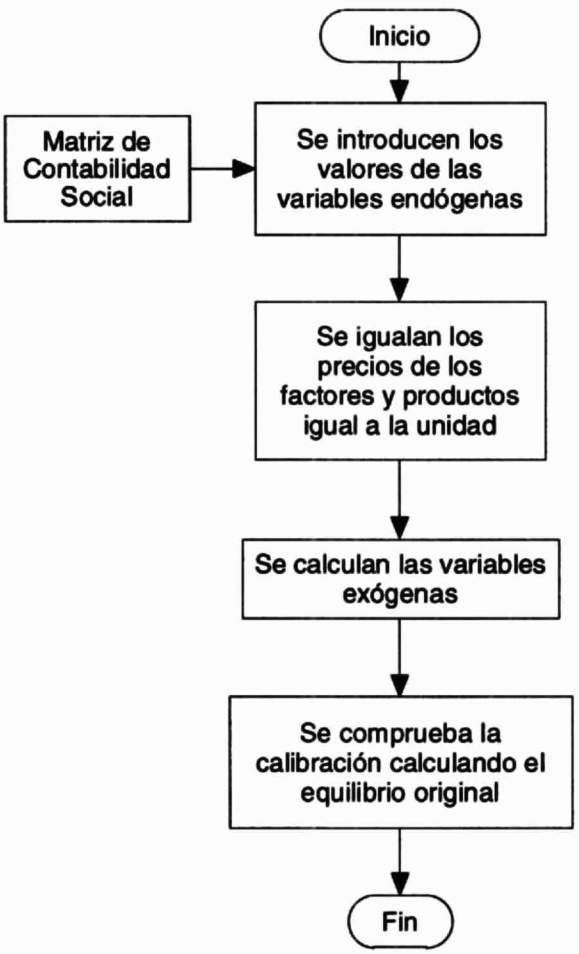
Apéndice A.1.

Algoritmo de solución de un modelo de equilibrio general aplicado



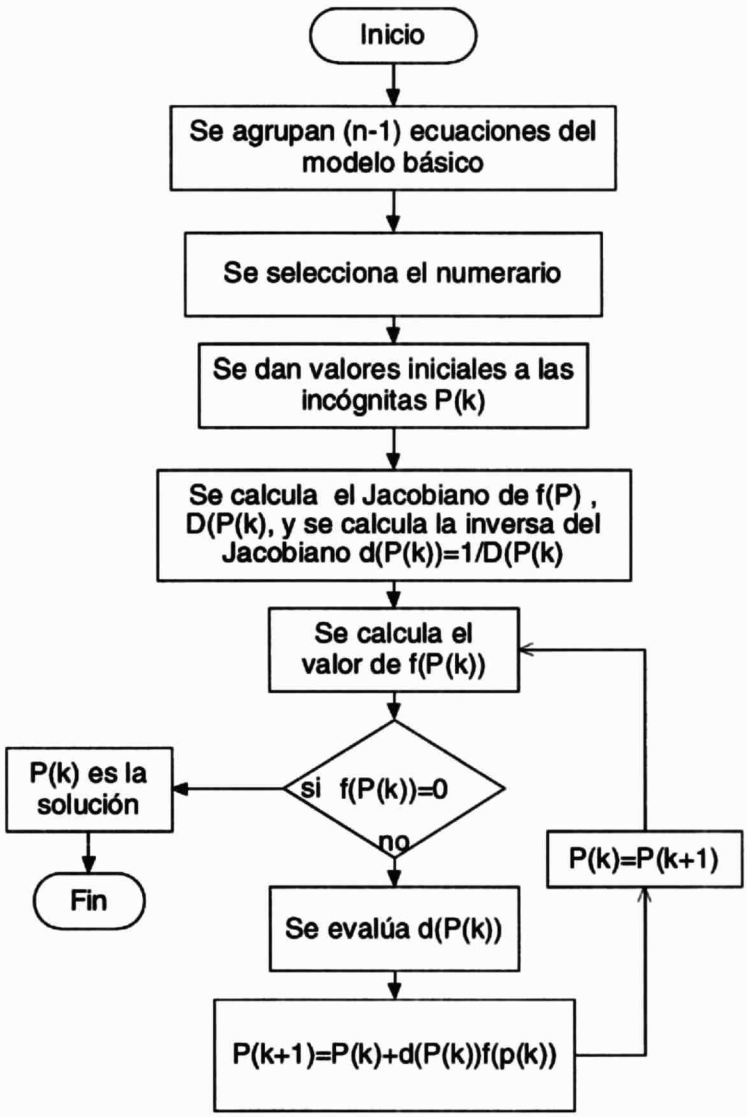
Apéndice A.2.

Algoritmo de calibración



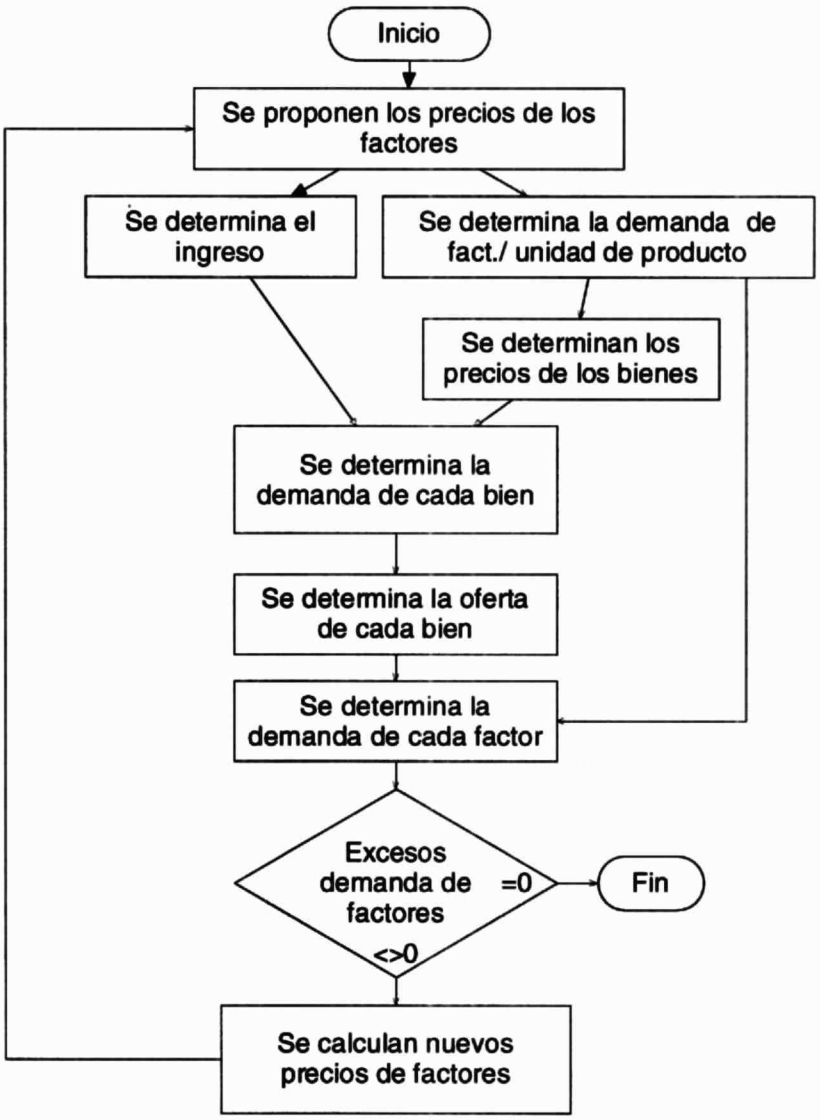
Apéndice A.3.

Estrategia y algoritmo de solución basados en el Método de Newton-Raphson



Apéndice A.4.

Estrategia basada en un proceso de Tâtonnement



Apéndice B.

Programa para la solución de un modelo de equilibrio general aplicado basado en el método de Newton-Raphson

■ Instrucciones

- **El programa esta diseñado para manejar N sectores y N factores**

- **El primer paso del programa solicita que se indique la cantidad de sectores y de factores.**

- **El segundo paso indica que se introduzcan los valores de una Matriz de Contabilidad Social, con el propósito de realizar la calibración del modelo y se puedan calcular los parámetros (variables exógenas).**

- **En el tercer paso el programa calcula el equilibrio original.**

- **En el cuarto paso el programa le pide al usuario indique el factor del cual la cantidad total variará, asi como el nuevo valor.**

- **En el quinto paso el programa inserta el nuevo valor en los parámetros y entonces calcula el nuevo equilibrio**

- **En el último paso, el programa imprime los equilibrios original y nuevo.**

- **Para encontrar los valores de equilibrio se utiliza una variación de la función Preconstruida FindRoot (la cual se basa en el método de Newton), llamada MyFindRoot escrita por Silvio Levy.**

- **Para lo cual es necesario que se guarde el programa MFRoot.m en el Subdirectorio AddOns\StandardPackages\Mega**

Inicio del programa

```
(* Se carga el programa para resolución de ecuaciones *)
(* Utilizando el método de Newton *)

<< Mega`MFRoot`
```

■ Sectores y Factores

```
(*Define el número de sectores y de factores*)

TSectors = Input["¿Cuántos sectores?"];
TFactors = Input["¿Cuántos factores?"];
```

■ Definición de Ecuaciones

```
defecuacionesofertademanda[nada_] :=

 ((*Fórmula de Función de Producción tipo Cobb-Douglas
 funcionproduccion[i]=A[i] ∏_{j=1}^{TFactors} L[i,j]^{b[i,j]}, i=1..TSectors,j=1,.,TFactors*)

 Do[funcionproduccion[i] =
    A[i] Product[L[i, j] ^b[i, j], {j, 1, TFactors}], {i, 1, TSectors}];

 (*Productividad marginal del factor j-
     ésimo del sector i-esimo prodmarginal[i,j]=
    δF[i]/δL[i,j]   i=1..TSectors,j=1,.,TFactors*)

 Do[prodmarginal[i, j] = D[funcionproduccion[i], L[i, j]],
   {i, 1, TSectors}, {j, 1, TFactors}];

  (*Función de costos C[i]=∑_{j=1}^{TFactors} w[j] L[i,j]  *)

  Do[costos[i] = Sum[w[j] L[i, j], {j, 1, TFactors}], {i, 1, TSectors}];

(* Función de utilidad, U=∏_{i=1}^{TSectors} X[i]^{a[i]},   i=1..TSectors *)

  U = Product[X[i] ^a[i], {i, 1, TSectors}];

   (* Utilidad Marginal del Bien X[i],
    utilmarginal[i]=δU/δX[i]), i=1..TSectors *)

  Do[utilmarginal[i] = D[U, X[i]], {i, 1, TSectors}];

  (*Obtiene las ecuaciones de p[i]/p[1] = utilmarginal[i]/utilmarginal[1],
   es decir que la tasa económica de sustitución entre el bien i y el 1 es
    igual a la tasa marginal de sustitución entre el bien i y el 1 *)

  tasamargsust =
   Table[p[i] / p[1] == utilmarginal[i] / utilmarginal[1], {i, 2, TSectors}];

  (*La restricción presupuestal es   I=∑_{i=1}^{TSectors} p[i] X[i]*)

  consumobienes = Sum[p[i] X[i], {i, 1, TSectors}];

  restpresupuestal = List[ingreso == consumobienes];

  (* Iguala la suma de exponentes a 1 *)

  sumaexponentesdemanda = Sum[a[i], {i, 1, TSectors}];
  ecelasticidaddemanda = List[1 == sumaexponentesdemanda];
```

```
(*Une las ecuaciones para formar las ecuaciones de demanda *)

ecuacionesdemanda = Join[tasamargsust, restpresupuestal];

ecuacionesdemandasust = Join[tasamargsust,  ecelasticidaddemanda];

(*Obtiene las ecuaciones de w[j]/w[1] = prodmarginal[i,j]/prodmarginal[i,1],
 es decir que la tasa económica de sustitución entre el factor j y el 1 es
  igual a la tasa técnica de sustitución entre el factor j y el 1 ,
 para el sector i=1,..,TSectors*)

tasatecnicasust = Flatten[Table[w[j] / w[1] ==
    prodmarginal[i, j] / prodmarginal[i, 1], {j, 2, TFactors}, {i, 1, TSectors}]];

(*Genera las ecuaciones de la parte de la produccion*)

produccion = Table[Y[i] == funcionproduccion[i], {i, 1, TSectors}];

(*Iguala la suma de exponentes a 1 *)

 Do[sumaexponentesoferta[i] = Sum[b[i, j], {j, 1, TFactors}], {i, 1, TSectors}];
ecelasticidadoferta = Table[1 == sumaexponentesoferta[i], {i, 1, TFactors}];

(*Une las ecuaciones para formar las ecuaciones de produccion *)

ecuacionesoferta = Join[tasatecnicasust, produccion]

)

defecuacionesofertademanda[1];
```

```
defecuacionesequilibrio[artificial_] := (

(*Identidad de ingreso de factores I=∑_{j=1}^{TFactors} w[j] Ltot[j] *)

  factores = Sum[w[j] Ltot[j], {j, 1, TFactors}];

  ecuacioningresofactores = List[ingreso == factores];

  (*Ecuación
     de ganancias extraordinarias de cero p[i] Y[i]=∑_{j=1}^{TFactors} w[j] L[i,j] *)

  ecuacionesganextcero = Table[p[i] Y[i] == costos[i], {i, 1, TSectors}];

  (* Ecuaciones de mercados en equilibrio X[i]=Y[i] , i=1,...,TFactors *)

  ecuacionesmercequilibrio = Table[X[i] == Y[i], {i, 1, TSectors}];

  (*Suma de factores por sectores Ltot[j]=∑_{j=1}^{TSectors} L[i,j] *)

  Do[sumafactorporsectores[j] = Sum[ L[i, j], {i, 1, TSectors}], {j, 1, TFactors}];

  ecuacionescantfactores =
   Table[Ltot[j] == sumafactorporsectores[j], {j, 1, TFactors}]
  );


defecuacionesequilibrio[1];
```

■ Calibración

```
(* Para la calibración se requiere:

 1. Introducir valores de las variables endógenas de acuerdo a una
    matriz de contabilidad social.
 2. Se suponen los precios de los productos y factores
    igual a la unidad.
 3. Se calculan los valores de las variables exógenas como:
    a. Coeficientes.
    b. Elasticidades.
    c. Cantidades totales de factores. *)


(* Se generan funciones para preguntar el valor de las variables
  endógenas a utilizar *)
```

```
codigoascii[numero_] := (

  bb = IntegerDigits[numero];
 (*Desplaza los números para obtener sus códigos ASCII *)
   valorutil = bb + 48;
 (*Obtiene como texto los caracteres asociados a esos códigos ASCII*)
  expon = FromCharacterCode[valorutil]);


generatexto[texto1_, texto2_, texto3_, texto4_, texto5_, texto6_] := (

 (*Separa cada texto en caracteres*)
  g1 = Characters[texto1];
  g2 = Characters[texto2];
  g3 = Characters[texto3];
  g4 = Characters[texto4];
  g5 = Characters[texto5];
  g6 = Characters[texto6];
(*Ahora los une todos en una cadena*)
  cadena = Join[g1, g2, g3, g4, g5, g6];
(* Ahora lo que hace es generar un texto junto *)
 textofinal = StringJoin[cadena]);



(* Define los textos adicionales *)

 cadenatexto1 = "Introduce el valor de la variable : ";
 cadenatexto2 = "L[ ";
  cadenatexto4 = " ,";
  cadenatexto6 = "]";



 Do [(cadenatexto3 = codigoascii[i];
      cadenatexto5 = codigoascii[j];
      generatexto[cadenatexto1, cadenatexto2,
   cadenatexto3, cadenatexto4, cadenatexto5, cadenatexto6];
   L[i, j] = Input[textofinal]), {i, 1, TSectors}, {j, 1, TFactors}]



cadenatexto2 = "X[  ";
cadenatexto4 = " ";
cadenatexto5 = " ";


Do [(cadenatexto3 = codigoascii[i];
     generatexto[cadenatexto1, cadenatexto2,
   cadenatexto3, cadenatexto4, cadenatexto5, cadenatexto6];
   X[i] = Input[textofinal]), {i, 1, TSectors}]
```

```
(* Ahora se asigna el valor de la X[i]
   introducida a Y[i] *)
Do[Y[i] = X[i], {i, 1, TSectors}];

(* Se le da valor a los precios de factores y productos igual a uno *)
Do[p[i] = 1, {i, 1, TSectors}];
Do[w[j] = 1, {j, 1, TFactors}];


(******* A Partir de aqui se calculan los valores de los parámetros********)

(* Se genera una lista de incognitas referente a las variables a[i]*)
incognitas1 = Table[a[i], {i, 1, TSectors}];

(*Se calculan los valores de los
  coeficientes a[i] utilizando la función preconstruida NSolve[] *)

alfas = Flatten[NSolve[ecuacionesdemandasust, incognitas1]];



ecuacionesauxiliares = Join[ tasatecnicasust, ecelasticidadoferta];

incognitas2 = Flatten[Table[b[i, j], {i, 1, TSectors}, {j, 1, TFactors}]];

(*Se calculan los valores de los
  coeficientes b[i,j] utilizando la función preconstruida NSolve[] *)

betas = Flatten[NSolve[ecuacionesauxiliares, incognitas2]];



(* Ahora se calculan los valores de los coeficientes A[i] *)

incognitas3 = Table[A[i], {i, 1, TSectors}];

(*Se calculan los valores de los
  coeficientes A[i] utilizando la función preconstruida NSolve[] *)

coeficientesA = Flatten[NSolve[produccion, incognitas3]] /. betas;



(* Por último se calculan los valores totales Ltot[j] *)
incognitas4 = Table[Ltot[j], {j, 1, TFactors}];
totalfactores = Flatten[NSolve[ecuacionescantfactores, incognitas4]];
```

```
(* Los valores de los
  parámetros después de la calibración son los siguientes *)

parametros = Join[coeficientesA, totalfactores, alfas, betas]
```

■ Cálculo del equilibrio original

```
(* Se borran los valores introducidos de las
  variables endógenas usando un equivalente de la funcion Clear[] *)

Borrar[artificial3_] := (Do[L[i, j] =. , {i, 1, TSectors}, {j, 1, TFactors}];
Do[X[i] =. , {i, 1, TSectors}];
Do[Y[i] =., {i, 1, TSectors}];
Do[p[i] =., {i, 1, TSectors}];
Do[w[j] =., {j, 1, TFactors}]);

Borrar[1]

(* Se generan valores iniciales *)


prec = 1. 10^-6;
iniciales[anad_] :=
 (  Do[Linicial[i, j] = Ltot[j] - prec, {j, 1, TFactors}, {i, 1, TSectors}];

(* Se le da formato a las variables con sus variables iniciales *)
(* para que puedan ser procesados por la máquina *)
Do[varL[i, j] = List[List[L[i, j], Linicial[i, j], 0, Ltot[j]]],
   {j, 1, TFactors}, {i, 1, TSectors}];
  Do[varCons[i] = List[List[X[i], 1, 0, 100000]], {i, 1, TSectors}];
  Do[varProd[i] = List[List[Y[i], 1, 0, 100000]], {i, 1, TSectors}];
   Do[varPrecio[i] = List[List[p[i], 1, 0, 100000]], {i, 1, TSectors}];
   Do[varwages[j] = List[List[w[j], 1, 0, 100000]], {j, 1, TFactors}];
   varIngreso = Flatten[List[List[ingreso, 1, 0, 100000]]])


iniciales[1];
```

```
(* Se asigna a w[1] como el numerario *)

w[1] = 1;

(* Se agrupan las variables y sus valores
  iniciales para formar una cadena llamada "variablesincognitas" que
  posteriormente se utilizara para calcular los valores de equilibrio *)

variables1 = Flatten[Table[varL[i, j], {i, 1, TSectors}, {j, 1, TFactors}]];
variables2 = Flatten[Table[varCons[i], {i, 1, TSectors}]];
variables3 = Flatten[Table[varProd[i], {i, 1, TSectors}]];
variables4 = Flatten[Table[varPrecio[i], {i, 1, TSectors}]];
variables5 = Flatten[Table[varwages[j], {j, 1, TFactors}]];

(* Eliminamos de las incógnitas al numerario, que en este caso es w[1] *)

variables5bis = Drop[variables5, {1, 4}];

(* Une todas las variables incognitas en la variable variablesincognitas *)

variablesincaux = Join[variables1, variables2,
   variables3, variables4, variables5bis, varIngreso] /. parametros;

variablesincognitas = Partition[variablesincaux, 4];
```

```
(***** Ahora se agrupan las ecuaciones *******)

(* Como la ley de Walras establece,
 el equilibrio en n-1 mercados implica el equilibrio en el n-esimo mercado*)
(* Por lo tanto no se considerará
  la última ecuación de las ecuaciones ecuacionescantfactores,
 las cuales son las ecuaciones de equilibrio en los mercados de factores *)

ecuacionescantfactaux = Drop[ecuacionescantfactores, -1];

(* Ahora
  se agruparan las ecuaciones en una variable llamada ecuacionesincognitas

  Tenemos una cantidad de 3*TSectors+
  TFactors+TSectors*TFactors, es decir que si
  se tuvieran TSectors=2 y TFactors=2,
 entonces se tendrían 12 ecuaciones con 12 incógnitas *)

ecuacionesincognitas = Join[ecuacionesdemanda,
   ecuacioningresofactores, ecuacionesoferta, ecuacionesganextcero,
   ecuacionesmercequilibrio, ecuacionescantfactaux] /. parametros;



(* Ahora se define una función para el cálculo del equilibrio *)

equilibrio[datos_, sotad_] := MyFindRoot @@ (Prepend[datos, sotad]);

(* Y se procede a calcular el equilibrio original *)

equilibriooriginal = equilibrio[variablesincognitas, ecuacionesincognitas] ;
```

■ Nuevo equilibrio

```
numerofactor = Input["¿Qué factor desea modificar?"];
nuevovalor = Input["Introduzca el nuevo valor"];

(*Sustituye el nuevo valor
  de la cantidad del factor en la variable de parametros*)

parametrosnuevos =
 ReplacePart[parametros, Ltot[numerofactor] -> nuevovalor, 2 + numerofactor]
```

```
variablesincauxnuevas = Join[variables1, variables2, variables3,
   variables4, variables5bis, varIngreso] /. parametrosnuevos;

variablesincognitasnuevas = Partition[variablesincauxnuevas, 4];

ecuacionesincognitasnuevas =
 Join[ecuacionesdemanda, ecuacioningresofactores, ecuacionesoferta,
   ecuacionesganextcero, ecuacionesmercequilibrio, ecuacionescantfactaux] /.
  parametrosnuevos;

(* Y se procede a calcular el equilibrio nuevo *)

equilibrionuevo =
 equilibrio[variablesincognitasnuevas, ecuacionesincognitasnuevas] ;
```

■ Comparación de los equilibrios

```
(* Se calculan las
  utilidades original y nueva y los ingresos original y nuevo *)
 Utilidadoriginal = U /. equilibriooriginal /. parametros;
Utilidadnueva = U /. equilibrionuevo /. parametrosnuevos;
ingresooriginal = consumobienes /. equilibriooriginal;
ingresonuevo = consumobienes /. equilibrionuevo;

variacioncompensada =
 (Utilidadnueva - Utilidadoriginal) * ingresonuevo / Utilidadnueva;

variacionequivalente =
 (Utilidadnueva - Utilidadoriginal) * ingresooriginal / Utilidadoriginal;
```

■ Impresion de resultados

```
Print["\n El equilibrio original es: \n", equilibriooriginal];
Print["\n El equilibrio nuevo es: \n", equilibrionuevo];
Print["\n La variación compensada es: ", variacioncompensada];
Print["\n La variación equivalente es: ", variacionequivalente];
```

Apéndice C.

Programa para la solución de un modelo de equilibrio general aplicado basado en el método de Tâtonnement

■ Instrucciones

- El programa esta diseñado para manejar N sectores y N factores
- El primer paso del programa solicita que se indique la cantidad de sectores y de factores.
- El segundo paso indica que se introduzcan los valores de una Matriz de Contabilidad Social, con el propósito de realizar la calibración del modelo y se puedan calcular los parámetros (variables exógenas).
- En el tercer paso el programa calcula el equilibrio original con el método de Tattonement.

Inicio del programa

■ Sectores y Factores

```
(*Define el número de sectores y de factores*)

TSectors = Input["¿Cuántos sectores?"];
TFactors = Input["¿Cuántos factores?"];
```

■ Definición de Ecuaciones

```
defecuacionesofertademanda[nada_] :=

 ((*Fórmula de Función de Producción tipo Cobb-Douglas
funcionproduccion[i]=A[i] ∏_{j=1}^{TFactors} L[i,j]^{b[i,j]}, i=1..TSectors,j=1,.,TFactors*)

Do[funcionproduccion[i] =
   A[i] Product[L[i, j] ^b[i, j], {j, 1, TFactors}], {i, 1, TSectors}];

(*Productividad marginal del factor j-ésimo del sector i-
    esimo prodmarginal[i,j]=δF[i]/δL[i,j]  i=1..TSectors,j=1,.,TFactors*)

Do[prodmarginal[i, j] = D[funcionproduccion[i], L[i, j]],
  {i, 1, TSectors}, {j, 1, TFactors}];

 (*Función de costos C[i]=∑_{j=1}^{TFactors} w[j] L[i,j] *)

 Do[costos[i] = Sum[w[j] L[i, j], {j, 1, TFactors}], {i, 1, TSectors}];

(* Función de utilidad, U=∏_{i=1}^{TSectors} X[i]^{a[i]},  i=1..TSectors *)

 U = Product[X[i] ^a[i], {i, 1, TSectors}];

  (* Utilidad Marginal del Bien X[i],
   utilmarginal[i]=δU/δX[i]), i=1..TSectors *)

 Do[utilmarginal[i] = D[U, X[i]], {i, 1, TSectors}];

 (*Obtiene las ecuaciones de p[i]/p[1] = utilmarginal[i]/utilmarginal[1],
  es decir que la tasa económica de sustitución entre el bien i y el 1 es
   igual a la tasa marginal de sustitución entre el bien i y el 1 *)

 tasamargsust =
  Table[p[i] / p[1] == utilmarginal[i] / utilmarginal[1], {i, 2, TSectors}];

 (*La restricción presupuestal es   I=∑_{i=1}^{TSectors} p[i] X[i]*)

 consumobienes = Sum[p[i] X[i], {i, 1, TSectors}];

 restpresupuestal = List[ingreso == consumobienes];

 (* Iguala la suma de exponentes a 1 *)

 sumaexponentesdemanda = Sum[a[i], {i, 1, TSectors}];
 ecelasticidaddemanda = List[1 == sumaexponentesdemanda];

 (*Une las ecuaciones para formar las ecuaciones de demanda *)
```

```
ecuacionesdemanda = Join[tasamargsust, restpresupuestal];

ecuacionesdemandasust = Join[tasamargsust,  ecelasticidaddemanda];

(*Obtiene las ecuaciones de w[j]/w[i] = prodmarginal[i,j]/prodmarginal[i,1],
 es decir que la tasa económica de sustitución entre el factor j y el 1 es
  igual a la tasa técnica de sustitución entre el factor j y el 1 ,
 para el sector i=1,..,TSectors*)

tasatecnicasust = Flatten[Table[w[j] / w[1] ==
    prodmarginal[i, j] / prodmarginal[i, 1], {j, 2, TFactors}, {i, 1, TSectors}]];

(*Genera las ecuaciones de la parte de la produccion*)

produccion = Table[Y[i] == funcionproduccion[i], {i, 1, TSectors}];

(*Iguala la suma de exponentes a 1 *)

 Do[sumaexponentesoferta[i] = Sum[b[i, j], {j, 1, TFactors}], {i, 1, TSectors}];
ecelasticidadoferta = Table[1 == sumaexponentesoferta[i], {i, 1, TFactors}];

(*Une las ecuaciones para formar las ecuaciones de produccion *)

ecuacionesoferta = Join[tasatecnicasust, produccion]

)

defecuacionesofertademanda[1];
```

```
defecuacionesequilibrio[artificial_] := (

(*Identidad de ingreso de factores I=∑_{j=1}^{TFactors} w[j] Ltot[j] *)

  factores = Sum[w[j] Ltot[j], {j, 1, TFactors}];

  ecuacioningresofactores = List[ingreso == factores];

  (*Ecuación
     de ganancias extraordinarias de cero p[i] Y[i]=∑_{j=1}^{TFactors} w[j] L[i,j] *)

  ecuacionesganextcero = Table[p[i] Y[i] == costos[i], {i, 1, TSectors}];

  (* Ecuaciones de mercados en equilibrio X[i]=Y[i] , i=1,...,TFactors *)

  ecuacionesmercequilibrio = Table[X[i] == Y[i], {i, 1, TSectors}];

  (*Suma de factores por sectores Ltot[j]=∑_{j=1}^{TSectors} L[i,j] *)

  Do[sumafactorporsectores[j] = Sum[ L[i, j], {i, 1, TSectors}], {j, 1, TFactors}];

  ecuacionescantfactores =
   Table[Ltot[j] == sumafactorporsectores[j], {j, 1, TFactors}]
  );


defecuacionesequilibrio[1];
```

■ Calibración

```
(* Para la calibración se requiere:

 1. Introducir valores de las variables endógenas de acuerdo a una
    matriz de contabilidad social.
 2. Se suponen los precios de los productos y factores
    igual a la unidad.
 3. Se calculan los valores de las variables exógenas como:
    a. Coeficientes.
    b. Elasticidades.
    c. Cantidades totales de factores. *)



(* Se generan funciones para preguntar el valor de las variables
  endógenas a utilizar *)
```

```
codigoascii[numero_] := (

  bb = IntegerDigits[numero];
 (*Desplaza los números para obtener sus códigos ASCII *)
  valorutil = bb + 48;
 (*Obtiene como texto los caracteres asociados a esos códigos ASCII*)
  expon = FromCharacterCode[valorutil]);


generatexto[texto1_, texto2_, texto3_, texto4_, texto5_, texto6_] := (

 (*Separa cada texto en caracteres*)
  g1 = Characters[texto1];
  g2 = Characters[texto2];
  g3 = Characters[texto3];
  g4 = Characters[texto4];
  g5 = Characters[texto5];
  g6 = Characters[texto6];
(*Ahora los une todos en una cadena*)
  cadena = Join[g1, g2, g3, g4, g5, g6];
(* Ahora lo que hace es generar un texto junto *)
 textofinal = StringJoin[cadena]);



(* Define los textos adicionales *)

 cadenatexto1 = "Introduce el valor de la variable : ";
 cadenatexto2 = "L[ ";
  cadenatexto4 = " ,";
  cadenatexto6 = "]";



 Do [(cadenatexto3 = codigoascii[i];
      cadenatexto5 = codigoascii[j];
      generatexto[cadenatexto1, cadenatexto2,
   cadenatexto3, cadenatexto4, cadenatexto5, cadenatexto6];
   L[i, j] = Input[textofinal]), {i, 1, TSectors}, {j, 1, TFactors}]



cadenatexto2 = "X[  ";
cadenatexto4 = " ";
cadenatexto5 = " ";


Do [(cadenatexto3 = codigoascii[i];
     generatexto[cadenatexto1, cadenatexto2,
   cadenatexto3, cadenatexto4, cadenatexto5, cadenatexto6];
   X[i] = Input[textofinal]), {i, 1, TSectors}]
```

```
(* Ahora se asigna el valor de la X[i]
   introducida a Y[i] *)
Do[Y[i] = X[i], {i, 1, TSectors}];

(* Se le da valor a los precios de factores y productos igual a uno *)
Do[p[i] = 1, {i, 1, TSectors}];
Do[w[j] = 1, {j, 1, TFactors}];


(******* A Partir de aqui se calculan los valores de los parámetros********)

(* Se genera una lista de incognitas referente a las variables a[i]*)
incognitas1 = Table[a[i], {i, 1, TSectors}];

(*Se calculan los valores de los
  coeficientes a[i] utilizando la función preconstruida NSolve[] *)

alfas = Flatten[NSolve[ecuacionesdemandasust, incognitas1]];



ecuacionesauxiliares = Join[ tasatecnicasust, ecelasticidadoferta];

incognitas2 = Flatten[Table[b[i, j], {i, 1, TSectors}, {j, 1, TFactors}]];

(*Se calculan los valores de los
  coeficientes b[i,j] utilizando la función preconstruida NSolve[] *)

betas = Flatten[NSolve[ecuacionesauxiliares, incognitas2]];



(* Ahora se calculan los valores de los coeficientes A[i] *)

incognitas3 = Table[A[i], {i, 1, TSectors}];

(*Se calculan los valores de los
  coeficientes A[i] utilizando la función preconstruida NSolve[] *)

coeficientesA = Flatten[NSolve[produccion, incognitas3]] /. betas;



(* Por último se calculan los valores totales Ltot[j] *)
incognitas4 = Table[Ltot[j], {j, 1, TFactors}];
totalfactores = Flatten[NSolve[ecuacionescantfactores, incognitas4]];


(* Los valores de los parámetros después de la calibración son los siguientes *)

parametros = Join[coeficientesA, totalfactores, alfas, betas]
```

■ Método de Tatonnement

```
(*Se define una función que permita borrar los valores de las variables
  endógenas llamada Borrar[]*)

Borrar[artificial3_] := (Do[L[i, j] =. , {i, 1, TSectors}, {j, 1, TFactors}];
Do[X[i] =. , {i, 1, TSectors}];
Do[Y[i] =., {i, 1, TSectors}];
Do[p[i] =., {i, 1, TSectors}];
Do[w[j] =., {j, 1, TFactors}]);

(* Se
  usa la función Borrar[] para eliminar los valores de las variables endógenas*)

Borrar[1]

varsalida = 1;

(* Se le dan valores iniciales a los precios de los factores *)

Do[w[j] = 2, {j, 1, TFactors}];

(* Genera una serie de instrucciones para generar un conjunto de soluciones
  que al final le asignaran el valor de 1 a las Y[i] esto es con
  objeto de que estos valores sean asignados a las variables L[i,j],
 debido a que en la fórmula de una Cobb-
  Douglas las ecuaciones de las demandas condicionadas tienen a la
   producción con exponente unitario y entonces estos resultados de
   las L[i,j] representarán las demandas condicionadas por producto *)

  ecuaciones = Flatten[Table[Y[i] == 1, {i, 1, TSectors}]];
  listaofertabienes = Flatten[Table[Y[i], {i, 1, TSectors}]];
  solucion = Flatten[Solve[ecuaciones, listaofertabienes]];

While[varsalida == 1,
 (

  (************  Primero se
    obtiene la demanda condicionada por producto  ***************)

   (* Se insertan los valores de los parámetros en las ecuacionesoferta
    y se le da el nombre de ecuacionesofertaconparametros *)

  ecuacionesofertaconparametros = ecuacionesoferta /. parametros;

  (* Se crea una lista de las incógnitas *)

   listademandasporproducto =
   Flatten[Table[L[i, j], {i, 1, TSectors}, {j, 1, TFactors}]];
```

```
 (* Obtiene los valores de la demanda condicionada por producto*)

demandascondporproducto = Flatten[
   Solve[ecuacionesofertaconparametros, listademandasporproducto]] /. solucion;

 (******************  Segundo
  se obtienen los precios  *************************)

(* Genera una lista de las ecuaciones con los valores calculados*)

ecuacionesprecios = ecuacionesganextcero /. demandascondporproducto /. solucion;

(*Genera una lista de las incógnitas*)

listaprecios = Flatten[Table[p[i], {i, 1, TSectors}]];

(*Obtiene los valores de los precios*)

precios = Flatten[Solve[ecuacionesprecios, listaprecios]];

(********************** Tercero
  se obtiene el ingreso *************************)

(* Obtiene el valor del ingreso *)

solingreso = Flatten[Solve[ecuacioningresofactores, {ingreso}]] /. parametros;

(**********************
  Cuarto se obtienen las demanda de bienes *************************)

(* Se genera una lista de las incógnitas de las demandas por bienes *)

listademandasbienes = Flatten[Table[X[i], {i, 1, TSectors}]];

(*Obtiene el valor de las demandas por bienes *)

demandas = Flatten[Solve[ecuacionesdemanda, listademandasbienes]] /.
    parametros /. solingreso /. precios;

(**********************
  Quinto se obtienen las ofertas de bienes *************************)

ofertas =
 Flatten[Solve[ecuacionesmercequilibrio, listaofertabienes]] /. demandas;

(********************** Sexto
  se obtienen las demandas de los factores ******************)

(* Se generan las ecuaciones donde se igualan las
  demandas de los factores al producto de las demandas de factores
```

```
 por producto multiplicadas por la cantidad de bienes producidos*)

ecuacionesdemandafactores = Flatten[
  Table[demfactor[i, j] == L[i, j] * Y[i], {i, 1, TSectors}, {j, 1, TFactors}]];

(* Se genera la lista de las incógnitas de demanda de factores *)

listademandafactores =
 Flatten[Table[demfactor[i, j], {i, 1, TSectors}, {j, 1, TFactors}]];

 (* Se obtiene el valor de las demandas de factores *)

 demandafactores =
  Flatten[Solve[ecuacionesdemandafactores, listademandafactores]] /. ofertas /.
   demandascondporproducto;

 (************************
   Séptimo se obtiene la demanda total por factor ****************)

  Do[
  Totaldemandafactor[j] = Sum[demfactor[i, j], {i, 1, TSectors}], {j, 1, TSectors}];

 (******************
   Octavo se obtienen los excesos de demanda de los factores ***********)

 Excesos = Table[excesodemanda[j] = Totaldemandafactor[j] - Ltot[j],
     {j, 1, TFactors}] /. demandafactores /. parametros;

 (*****************************************************************************
   ************)

  (* Se establece el parámetro de precisión *)

 prec = 0.001

  (* Se establece el parámetro de ajuste w_{j,1}=w_j+θ (Exceso de demanda) *)

 Theta = 0.01;

  (* Se verifican los excesos de
   demanda y entonces se ajustan los precios de los factores *)

 For[j = 1, j <= TFactors, j++, If[Excesos[[j]] <= prec , Continue[],
    (w[j] += Theta * Excesos[[j]];
     Break[])]];
 Print["\ndemandascondporproducto : ", demandascondporproducto, "\nprecios: ",
  precios, "\nsolingreso: ", solingreso, "\ndemandas: ", demandas, "\nofertas: ",
  ofertas, "\ndemandafactores: ", demandafactores, "\nexcesos: ", Excesos];
 If[j == TFactors + 1, varsalida = 0]

 )]
```

Apéndice D

Programa para la solución de un Sistema de Ecuaciones Basado en el Método de Newton – Raphson

```
(* :Title: MyFindRoot *)
(* :Author: Silvio Levy *)

(* :Summary: Searches for a numerical solution of the equations with
more assurance than the built-in FindRoot function. *)

(* :Context: MyFindRoot` *)

(* :Package Version: 1 *)

(* :Copyright: Copyright 1991 Silvio Levy. *)

(* :Mathematica Versions: 1.2 *)

(* :History:
  Separated from the package GeneralEquilibrium.m, which were supplied
  in the electronic supplement of The Mathematica Journal, Volume 1,
  Issue 3, Winter 1991.  GeneralEquilibrium.m was written by Asahi
  Noguchi in August 1990, modified by Silvio Levy in February/March
  1991.
*)

(* :Discussion:
MyFindRoot is slower than FindRoot (though not by a large factor).
It always uses approximations to the derivative, rather than computing
derivatives symbolically.  Therefore the specification of variables
is of the form {x,x0} or {x,x0,xmin,xmax}, and never {x,{x0,x1}} as
with FindRoot.
*)

BeginPackage["MyFindRoot`"]

MyFindRoot::usage = "\n
  MyFindRoot is variant of FindRoot that doesn't give up when\n
  it lands out of bounds; rather, it uses various heuristics\n
  to try to recover and continue the search for a solution.\n
  It is used like FindRoot."

Begin["`private`"]

MyFindRoot::tomanyiter = "too many iterations."
MyFindRoot::wrongdim = "wrong number of equations or variables."
MyFindRoot::nonnum = "non-numerical result of equations."

SeedRandom[0]
Off[Plot::notnum]
Off[ParametricPlot::notpoint]

MyFindRoot[oeqs_,varsandopts__] :=
```

```
Block[
  {dim,vars,opts,ag,maxiter,eqs=oeqs,
    eps=.5 10^-6,derivs,image,i=0,j,sol,t,names,vals,code},
  vars=Cases[{varsandopts},_List];
  opts=Complement[{varsandopts},vars];
  ag=AccuracyGoal/.opts/.Options[FindRoot];
  If[SameQ[ag,Automatic], ag=eps, ag=10^-ag];
  maxiter=MaxIterations/.opts/.Options[FindRoot];
  If[!SameQ[Head[eqs],List],eqs=List[eqs]];
  eqs=eqs/.Equal[x_,y_]->x-y;
  dim=Length[vars];
  If[dim!=Length[eqs], Message[MyFindRoot::wrongdim]; Return[Null]];
  names=First /@ vars;
  vals=First/@(Rest/@ vars);
  bounds=Rest/@(Rest /@ vars);

  While[True,
    rules=Thread[names->vals];
    image=N[eqs/.rules];
    If[!(And@@(NumberQ/@image)), Message[ MyFindRoot::nonnum]; Return[Null]];
    If[Max[Abs[image]]<ag, Return[rules]];
    If[i==maxiter, Message[MyFindRoot::tomanyiter]; Return[Null]];
    derivs=Table[N[eqs/. MapAt[(#+eps)&,rules,{j,2}]]-image,{j,1,dim}];
    sol=vals - image.Inverse[derivs/eps];
    t=adjust[Min[timeout /@ Transpose[{sol,vals,bounds}]]];
    vals=Thread[jolt[t sol + (1-t)vals,bounds]];
    i++;
  ]
]

timeout[{newval_,curval_,{}}] := Infinity
timeout[{newval_,curval_,{lo_,hi_}}] :=
  If[newval<curval,(lo-curval)/(newval-curval),(curval-hi)/(curval-newval)]

adjust[t_] := If[t>4, 1, 2t/(4+t)]

jolt[val_?NumberQ,{}] := val
jolt[val_?NumberQ,{lo_,hi_}] :=
  If[val-lo< .001 || hi-val < .001, Random[Real,{.25(3lo+hi),.25(lo+3hi)}],val]

End[]
EndPackage[]
```

Apéndice E.

METODO DE NEWTON-RAPHSON[18]

Considérese un conjunto de funciones no lineales :

$$f_i(P_1, P_2, ..., P_n)$$

La cual en términos matriciales, se expresa como $f(P)$, donde P es el vector de variables y f es el vector de las funciones.
Si P es la solución de $f(P)$ se tiene que:

$$f(P)=0 \qquad \text{(A.1)}$$

En general, cualquier procedimiento iterativo para resolver el conjunto de ecuaciones se puede escribir como:

$$P^{(k+1)}=P^{(k)}+\alpha^{(k)}d^{(k)} \qquad \text{(A.2)}$$

Donde k se refiere a la iteración, $d^{(k)}$ es el vector de dirección y $\alpha^{(k)}$ es un escalar que indica el tamaño del paso a tomar en la dirección $d^{(k)}$.

En el algoritmo de Tâtonnement, el vector de dirección esta dado simplemente por el signo de $f(P)$, y el tamaño del paso se ajusta por el usuario para cada modelo en particular. En los algoritmos con Jacobianos, el vector de dirección depende de la matriz de derivadas de las funciones $f(P)$. Definimos la matriz de derivadas de las funciones como D:

$$D_{ij} = \frac{\partial f_i}{\partial P_j} \qquad \text{(A.3)}$$

Una clásica aproximación para resolver la ecuación (A.1) es el uso de la serie de Taylor para $f(P)$:

$$f(P) \approx f(P^{(k)})+D(P^{(k)})(P-P^{(k)}) \qquad \text{(A.4)}$$

Estableciendo $f(P)=0$ y resolviendo para $P=P^{(k)}$ se obtiene:

$$P^{(k+1)}=P^{(k)}-D^{-1} f(P^{(k)}) \qquad \text{(A.5)}$$

Este es el método de Newton o Newton-Raphson, con el vector de dirección d dado por $D^{-1}f$ y el tamaño del paso α igual a 1.

---

[18] Tomado de Dervis, De Melo y Robinson (1982).

Bibliografía.

Bergman, L , Dale, J Y Zalai, E. (1990). *General Equilibrium Modeling and Economic Policy Analysis*, Oxford, Basil Blackwell, Inc.

Chiang, A. (1993). *Métodos Fundamentales de Economía Matemática*. México, Mc Graw Hill.

Dervis,K., De Melo, J. Y Robinson,S. (1982). *General equilibrium models for development policy*, Cambridge, Cambridge University Press.

Debreu, G. (1959) Theory of Value. An Axiomatic Analysis of Economic Equilibrium. New York, Wiley.

Gaylord, R, Kamin,S Y Wellin, P. (1993). *Introduction to Programming with Mathematica*, California, Telos, Springer-Verlag.

Kehoe, P Y Kehoe, T. (1994). "A primer on Static applied General Equilibrium Models", *Quarterly Review*, Federal Reserve Bank of Minneapolis, vol. 18, no. 2, 2-16.

---------- (1994). "Capturing NAFTA´s Impact With Applied General Equilibrium Models", *Quarterly Review*, Federal Reserve Bank of Minneapolis, vol. 18, no. 2, 17-34.

Harberger, A. (1962). "The incidence of the Corporation Income Tax", *Journal of Political Economy*, vol.70, no.3, 215-240.

Maeder, R. (1997). *Programming in Mathematica*, Addison Wesley Longman, Inc.

Mas-Colell, A. , Whinston, M. Y Green, J. (1995). *Microeconomic Theory*. Oxford University Press.

Noguchi, A. (1991). "The Two-Sector General Equilibrium Model: Numerical and Graphical Representations of an Economy", *The Mathematica Journal*, vol.1, Issue.3 Winter, 96-103.

---------- (1992), "General Equilibrium Models", *Economic and Financial Modeling with Mathematica*, Varian Hal, Telos, Springer-Verlag. 105-123.

Pérez, A. Y González A. (2000). "Manual de Elaboración de Modelos de Equilibrio General aplicados" versión preliminar, Documento de Trabajo, Instituto Tecnológico de Estudios Superiores de Monterrey –CCM, México.

Scarf, H. Y Shoven, J. (1984). *Applied general equilibrium analysis*. Cambridge, Cambridge University Press.

Shoven, J. Y Whalley, J. (1984). "Applied General-Equilibrium Models of Taxation and International Trade: An Introduction and Survey", *Journal of Economic Literature*, vol. XXII, 1007-1051.

Simon, C Y Blume, L (1994). *Mathematics for Economists*, New York, W.W. Norton & Company, Inc., New York.

Srinivasan, T. Y Whalley J. (1986). *General Equilibrium Trade Policy Modeling*, The MIT Press.

Urzua, C. (1994). "Resuelve: A Gauss program for solving computable general equilibrium and disequilibrium models", *Estudios Económicos*, El Colegio de México, vol.9, núm. 2, 273-294.

Varian, H. (1992). *Microeconomic Analysis*, W.W. New York, Norton & Company.

Varian, H. (1996). *Computational Economics and Finance*, Modeling and analysis with Mathematica. California, Telos, The Electronic Library of Science.