Novedades de software/New softwares

UNA IMPLEMENTACION DEL AJUSTE DE DATOS MEDIANTE MINIMOS CUADRADOS GENERALIZADOS NO LINEALES PARA FUNCIONES VECTORIALES

Jorge Lemagne Pérez[1],
Departamento de Matemática Aplicada Facultad de Matemática y Computación, Universidad de La Habana

ABSTRACT

In this communication, a critical survey mainly on generalized least squares (GLS) software is made, and afterwards, to help to overcome the software limitations (according to our objectives) an algorithm for vector valued data fitting by GLS is presented. Moreover, it is shown another algorithm to estimate the variance-covariance matrix that corresponds to vectorized observation matrix, with different options for users. Both algorithms were implemented on MATLAB Version 7.3.

KEYWORDS: Generalized least squares, multivariate methods, data fitting, numerical methods software, computer implementation

MSC 93E24

RESUMEN

En esta comunicación se hace una revisión crítica del software existente sobre mínimos cuadrados generalizados (MCG, no lineales en general) principalmente, y posteriormente, para contribuir a superar las limitaciones existentes en dicho software (de acuerdo con nuestros objetivos) se presenta un algoritmo para resolver el problema de ajuste de datos (AD) mediante MCG para funciones vectoriales (FV). Además, se muestra otro algoritmo para estimar la matriz de varianza y covarianza correspondiente a la matriz de observaciones vectorizada, con diferentes opciones para el usuario. Ambos algoritmos fueron implementados en MATLAB Versión 7.3.

1. RECORDATORIO

En el Artículo 1 "Una presentación de los mínimos cuadrados generalizados, y en particular, para funciones vectoriales" (A1) fueron señaladas varias limitaciones existentes en la bibliografía sobre mínimos cuadrados, y más específicamente sobre AD mediante MCG para FV (abreviadamente AD_MCG_FV), en que se aproximan funciones

$$f : \Re^p \to \Re^q,\ p \geq 1,\ q \geq 1 \tag{1.1}$$

Dichas limitaciones nos motivaron a formalizar el planteamiento de dicho problema.

Mínimos cuadrados generalizados

Recordemos primeramente en qué consiste el problema de MCG (para más detalles, ver A1):

Sea $r_k : \Re^{n+1} \to \Re$, función no lineal (se sobreentiende en general) $k = 0, \ldots, N$.

$c \in \Re^{n+1}$, y denotaremos este vector como $c = (c_j)_{j=0,\ldots,n}$ (1.2)

Consideremos ahora la función vectorial

[1] lemagne@matcom.uh.cu

$$r(c) = \left(r_k(c) \right)_{k=0,\ldots,N}$$

y la matriz $W = \left(w_{ik} \right)_{i,k=0,\ldots,N}$, que supondremos que es simétrica y definida no negativa.

Se quiere entonces:

$$\text{Minimizar } E(c) \; : \; E(c) = r^T(c) W r(c) \tag{1.3}$$

Los parámetros c_j deben ser determinados para alcanzar dicho objetivo.

El problema anteriormente formulado es el de MCG**.** También se vio en A1 que para resolverlo se podía utilizar (al menos en teoría) la vía del sistema de ecuaciones normales (SEN, definición usada en un sentido más general), que concretamente es el siguiente:

$$J^T(c) W r(c) = 0 \quad : \quad J(c) = \left(\frac{\partial}{\partial c_j} r_k(c) \right)_{\substack{k=0,\ldots,N \\ j=0,\ldots,n}} \tag{1.4}$$

es la matriz jacobiana de $r(c)$.

$J(c)$ es una matriz de $N+1$ filas y $n+1$ columnas. ■

Ajuste de datos mediante mínimos cuadrados generalizados para funciones vectoriales

Recordemos además en qué consiste el AD_MCG_FV:

Sea $f : \Re^p \to \Re^q$, una función desconocida en la práctica $(p \geq 1, q \geq 1)$; $x_k \in \Re^p$; $f_k \in \Re^q$ y se conoce empíricamente, siendo f_k una aproximación al vector desconocido $f(x_k)$, $(k = 0,\ldots,N)$. $F : \Re^p \to \Re^q$ es una función de aproximación a f, $F(x) = F(x;\, c)$ donde c fue definido en (1.2). De acuerdo con el párrafo anterior, sea f_{kL} la componente L-sima del vector f_k. Además, por ser F una función de $\Re^p$ en $\Re^q$, esta puede descomponerse en las funciones $F_L : \Re^p \to \Re$, $L = 1,\ldots,q$. Adoptemos las notaciones:

$$r_{kL}(c) = f_{kL} - F_L(x_k; c)$$
$$R_L(c) = \left(r_{kL}(c) \right)_{k=0,\ldots,N}$$

Ahora hagamos

$$r = \begin{pmatrix} R_1(c) \\ R_2(c) \\ \vdots \\ R_q(c) \end{pmatrix}$$

y sea W una matriz cuadrada de orden $(N+1)\, q$, simétrica y definida no negativa.

Entonces, el problema de **AD mediante MCG para FV** (AD_MCG_FV) consiste en, con las suposiciones anteriores, realizar (1.3).

Aunque en la definición de AD_MCG_FV puede tomarse libremente cualquier W que cumpla las condiciones generales especificadas, en la práctica es usual hacer $W = V^+$, o sea la pseudoinversa de V (Gill et al [1990]), donde V es la matriz de varianza y covarianza de la matriz de observaciones vectorizada.

Inconveniencias de los mínimos cuadrados ordinarios

Ya en A1 señalamos la inconveniencia de aplicar Mínimos Cuadrados Ordinarios (MCO) cuando hay correlación en las observaciones. Para ahondar con respecto a este asunto, Chatterjee and Hadi [2006] explican que, cuando las observaciones tienen un orden secuencial natural, utilizamos el término "auto

correlación" para referirnos a la correlación, y que su presencia tiene varios efectos en el análisis del problema. Mencionemos dos de estos efectos:

1. Los estimados mínimos cuadráticos de los coeficientes de regresión son insesgados, pero no son eficientes, en el sentido que ya no tienen la varianza mínima.
2. Los estimados de σ^2 y de los errores estándar de los coeficientes de regresión podrían ser mucho menores de lo que en realidad son, dando una falsa impresión de exactitud.

2. REVISION DEL SOFTWARE[2]

Después del recordatorio matemático del epígrafe anterior, pasemos a analizar el software para resolver el problema de MCG, existente en la bibliografía consultada (tal como se había anunciado en A1) para determinar su utilidad, de acuerdo con nuestros objetivos. En aras de la completitud, comencemos pues por

AJUSTE DE DATOS MEDIANTE MINIMOS CUADRADOS PONDERADOS

Hager [1988] hace una reseña al paquete de subrutinas MINPACK-1, en donde se plantea el problema de MC no lineales (sin hacer referencia a función empírica).

Dentro de este grupo pudiéramos incluir al programa NLREG, para análisis de regresión no lineal (Sherrod [2003]) y a CurveExpert 1.3, sistema de ajuste de curvas para Windows (su autor es Hyams [2003]) que permite considerar ponderación, aunque $p = 1$.

El caso lineal naturalmente aparece tratado con mayor frecuencia, pero sin la suficiente generalidad y sistematización. Mencionaremos aquí al paquete estadístico compacto DataLab (Lohninger [2008]) ya que sólo admite unos pocos modelos no lineales; permite realizar regresión simple, $p = q = 1$ y no se hace ponderación.

Ajuste de datos mediante minimos cuadrados generalizados

Con el enfoque estadístico correspondiente, Zanella et al [2000] señalan lo siguiente:

"…nos imaginamos de qué manera debe ser enriquecido un manual corriente de control estadístico de la calidad para incluir métodos multivariados." Los mismos "representan una contraparte de los métodos univariados; creemos que estos últimos son las herramientas más populares para control estadístico y mejoramiento de procesos de producción usadas en la industria" (y con las que "los técnicos ya están bien familiarizados"). "Quizás debido a que hemos optado por tratar de establecer una armazón paralela entre ambos tipos de resultados, y a que los métodos multivariados aún no han sido suficientemente estudiados ni tratados, no hemos encontrado problemas que pudieran ser considerados como sustancialmente multivariados; esto trae como consecuencia que a menudo se han usado métodos estadísticos inapropiados en el caso univariado." "Quizás se necesitará algún software adicional, más orientado específicamente a las aplicaciones del análisis estadístico multivariado en la industria y en el control de procesos."

Caso no linealNCSS es un moderno sistema para el trabajo estadístico. El mismo contiene algunos procedimientos para el ajuste lineal de datos que permiten introducir varias variables dependientes, pero en este caso, analiza cada una de ellas *separadamente*. En el caso no lineal, con un amplio menú de funciones de aproximación, sólo se brinda la posibilidad de trabajar con *una* variable dependiente (véase Hintze [2007]).

En Smith [1993] aparece un algoritmo para MCG cuyo código en FORTRAN recibe el nombre de GLSMOD; aunque este puede utilizarse para resolver problemas de tamaño pequeño, presenta los siguientes inconvenientes:

1. Su información de entrada requiere de un gran trabajo preparatorio (este aspecto en particular fue señalado por el propio Smith), parte de esa información es la matriz de covarianza correspondiente al vector inicial de parámetros $c^{(0)}$, y la imagen de $c^{(0)}$ por la función que

[2] Además de las referencias del §7, la búsqueda de más de 140 publicaciones, registradas en A1, §9 ("Bibliografía"), resultó también de utilidad para este trabajo.

asigna a cada vector de parámetros c el vector formado por la aproximación a cada una de las observaciones. También es necesario suministrarle V, la matriz de varianza y covarianza de las observaciones.

2. $p = 1$.
3. Se exige que V sea definida positiva.

En realidad R no es un paquete estadístico, sino un ambiente de programación estadístico (UCLA, no aparece fecha). En Hengl [2007] se dice que "Muchos usuarios de R creen que en Estadística no hay mucho que R no pueda hacer."

En este ambiente se pueden ajustar datos (*no* vectoriales) mediante modelos no lineales. Se permite que los errores estén correlacionados y tengan varianzas distintas (ver R [2008]).

Caso lineal
Dentro de este caso (MCGL) incluimos a MathWorks [2004], que ajusta datos a un modelo (lineal o red neural), y admite la inclusión de matriz de covarianza.

También tenemos a WOMBAT, un paquete de software para análisis genético cuantitativo; este supone que el modelo es lineal y que las matrices de covarianza tienen rango completo (véase Meyer [2007]).

Ajuste de datos mediante minimos cuadrados generalizados para funciones vectoriales
El ambiente de programación interactivo S aborda la aproximación, para mediciones repetidas ("serial data"), cuando hay "multirespuestas" (multivariada con varias características), en este último caso realiza la regresión mediante splines, restringida a lineal más allá de los nodos exteriores (vea Alzola and Harrell [2002]). Además, el capítulo 8 de esa publicación trata sobre funciones suministradas por S para regresión lineal múltiple. [3]

SEM (Structural Equation Modeling) es una familia de técnicas estadísticas (LISREL, AMOS y EQS son 3 paquetes populares para realizar SEM). En esta última $q \geq 1$, y aunque ─al igual que en regresión─ es posible añadirle al modelo transformaciones no lineales de la variable original como exponencial o logarítmica, se supone que las relaciones entre las variables son lineales (vea Structural [2002] y Scientific Software International [2003]).

El potente software SAS permite el análisis de modelos en los cuales la relación entre las variables viene expresada mediante un sistema de una o más ecuaciones no lineales; la función empírica puede ser vectorial. La estimación mediante MCO no lineales se realiza iterativamente mediante una linealización del modelo con respecto a los parámetros.

En el método de MCG que proponemos en esta investigación no se utilizan variables instrumentales (SAS [2008]). En correspondencia, veamos los métodos de SAS para estimación de parámetros que no requieren de instrumentos, aparte de OLS (que es para MCO). Los mismos son: ITOLS, SUR, ITSUR y FIML.

En ITOLS, la matriz de covarianza de orden $(N + 1)q$ especificada en la función objetivo es diagonal simplemente. Sobre los restantes métodos, haremos más adelante un comentario. Por el momento, digamos que, acerca de FIML ("Full Information Maximum Likelihood") o máxima verosimilitud con información completa, la propia publicación señala que:

1. FIML supone que los errores de las ecuaciones tienen una distribución normal multivariada. Si los errores no están normalmente distribuidos, el método FIML pudiera producir pobres resultados.
2. El mismo es costoso desde el punto de vista computacional.

Limitaciones que pueden existir en la matriz de varianza y covarianza

[3] R está basado en S, lenguaje anterior de dominio público (Vinod [no aparece fecha]).

En los métodos SUR e ITSUR de SAS, la matriz de varianza y covarianza V tiene la misma forma que en Gallant [2000], Cué et al [1985], Marangoni [1999], Multiresponse [2003] y Saha [2003], ya explicada en A1, es decir:

$$V = \Sigma \otimes I \tag{2.1}$$

Como sabemos, esto trae como consecuencia dos restricciones:

1. Se consideran correlaciones *sólo* entre características de una misma observación o "individuo".[4]
2. Dadas dos características determinadas, las covarianzas correspondientes a cada uno de los individuos son todas *iguales*.

Similarmente, aunque –como hemos dicho– en SAS [2008] la función empírica puede ser vectorial, la función objetivo del método FIML involucra a Σ, que es la matriz $q \times q$ que representa simplemente las varianzas y covarianzas entre las q características, sin tomarse en cuenta las diferencias entre los individuos.

Como puede apreciarse, el software que se reporta en la bibliografía consultada no resulta adecuado para la resolución automática del problema general de AD_MCG_FV que formulamos en el §1, por tener las limitaciones que se han señalado.

Por lo tanto, tal como se había anunciado en Al, se presentará una implementación de dicho ajuste.

3. ALGORITMO PARA EL AJUSTE DE DATOS

Fundamento

El fundamento esencial del algoritmo realizado se explica a continuación. En (1.3), si $W = V^+$ (ver Gill et al [1990]):

$$\begin{aligned} E = r^T V^+ r &= r^T \left(U \Lambda U^T\right)^+ r \text{ , por la descomposición espectral de } V \\ &= r^T \left(U \Lambda^+ U^T\right) r = r^T \left(U D U^T\right) r \end{aligned} \tag{3.1}$$

Como V tiene l valores propios nulos $(l \geq 0)$, podemos decir entonces que:

$$E = r^T \left(U\_D\_U\_^T\right) r$$

donde $U\_$ es el resultado de eliminar en U los vectores propios columnas correspondientes a los valores propios nulos, y $D\_$ es una nueva matriz cuadrada diagonal que resulta de haber eliminado las l filas y columnas correspondientes a los mismos. Ahora bien:

$$E = s^T s \tag{3.2}$$

donde $s = D\_^{1/2} U\_^T r$

Para $r = f - F$,

$$s = \left(U\_D\_^{1/2}\right)^T f - \left(U\_D\_^{1/2}\right)^T F \tag{3.3}$$

Minimizar E en (3.2) es el problema de MCO, y para resolverlo utilizaremos la función "lsqcurvefit", que es suministrada por MATLAB, y realiza AD mediante MCO para FV.
Otras cuestiones relativas al algoritmo serán justificadas posteriormente.

Pasos esenciales del subprograma

El algoritmo fue programado en MATLAB Versión 7.3 mediante la función "MCG_Aju_dat_fun_vec".
Los pasos principales de este subprograma son los siguientes:

1. Asignación de los parámetros verdaderos de entrada: Valores_var_indep (valores de la variable independiente), Observaciones (valores de la variable dependiente), aprox_inicial (aproximación

[4] Esto también se aplica al ambiente de programación S.

inicial) y V (matriz de varianza y covarianza). Además el usuario debe suministrar las componentes de la función vectorial de aproximación F y sus matrices jacobianas, es decir F1, J1, F2, J2,... (las matrices jacobianas son opcionales).

2. Mediante la descomposición espectral de V, asignar a val_prop sus valores propios , y a V los vectores propios.[5]
3. Ordenar la información anterior de acuerdo con los valores propios, de mayor a menor.
4. Cálculo de "rango" (rango de la matriz original V) y reducción de la cantidad de valores y vectores propios.
5. Hallar el recíproco de cada valor propio.
   (Este paso es básico en el cálculo de la pseudoinversa, este no se realiza explícitamente).
6. Cálculo de $D\_^{1/2}U\_^{T}$ en (3.2) y asignar el resultado a V.[6]
7. Vectorizar "Observaciones".
8. Hacer Observaciones=V* Observaciones.
   (Ahora "Observaciones" tiene solo "rango" elementos. Este es íntegramente el primer término de s en (3.3) y será uno de los parámetros verdaderos en "lsqcurvefit").
9. Llamado a la función "lsqcurvefit".
   Los parámetros de entrada son: @Fun[7], aprox_inicial, Valores_var_indep, Observaciones, opciones[8]. La función "Fun", que tiene como objetivo principal suministrar el vector columna de aproximación (o sea, F en (3.3)), se explicará en el siguiente sub-epígrafe.
   Los parámetros de salida son: vec_param (vector de parámetros, con los valores de las incógnitas), norma_resid_al_cuad (norma del residual al cuadrado), menos_residual (residual cambiado de signo), indicador (del criterio de parada que se cumplió), otras_informac[9].
10. Conjuntamente con los parámetros de salida anteriores, se brinda el valor de "rango" en el parámetro "rango_V", y se pasa el control al programa que invocó a MCG_Aju_dat_fun_vec.

En el paso 3 del algoritmo anterior, la ordenación de los valores y vectores propios puede justificarse como sigue:[10]

En (3.1) $V = U\Lambda U^T$. Para cambiar el orden en que aparecen los valores propios en Λ, hacemos el cambio de variable $P^T\Lambda P = \Lambda_1$, donde Λ_1 contiene los valores propios en el orden deseado y P es una matriz de permutación tomada convenientemente. Entonces:

$$V = UP\Lambda_1 P^T U^T = U_1\Lambda_1 U_1^T$$

por lo que –intercambiando también en U las columnas– se mantiene la igualdad.[11] ■

Cálculo del vector de aproximación F

La función "Fun" es invocada por lsqcurvefit; y –como se dijo anteriormente– tiene como objetivo principal suministrar el vector columna de aproximación, es decir, F en (3.3)).

Los parámetros de entrada de Fun son: vec_param y Valores_var_indep.

Los de salida son: F (vector columna de aproximación) y J (matriz jacobiana de F). El algoritmo de cálculo de los mismos puede resumirse como sigue:

1. Iniciar F y J como vectores columnas vacíos.
2. Para característica desde 1 hasta cant_caract[12], repetir:
   2.1 Evaluar en (vec_param,Valores_var_indep) la función y la matriz jacobiana correspondientes a "característica", de acuerdo con las "functions" definidas por el usuario.

[5] Esta descomposición espectral se realiza a su vez mediante un pequeño subprograma. Para economizar espacio, los vectores propios sustituyen a V; debemos tener en cuenta que, desde el punto de vista de la memoria que requiere, V es una matriz costosa.
[6] Tampoco se realiza directamente, en aras de la eficiencia.
[7] Puntero a "Fun"
[8] Para indicar si se brindan o no matrices jacobianas.
[9] Informaciones sobre el algoritmo utilizado y el proceso iterativo, brindadas por "lsqcurvefit" (ver ejemplo en §5).
[10] Recordemos que, como la matriz es simétrica, todos sus valores propios son reales.
[11] Este razonamiento es válido cualquiera sea el orden deseado; en el caso particular de mayor a menor se aplica implícitamente cuando hacemos en *U* el "mismo cambio" que en Λ.
[12] cantidad de características, que es igual a *q* en (1.1).

2.2 Expandir F y J, respectivamente, con las evaluaciones del párrafo anterior, agregando estos elementos en los vectores columnas.

(Para cada invocación a Fun, este ciclo evalúa las funciones F1, J1, F2, J2,..., y va expandiendo F y J hasta completar la formación de ambos).

3. Hacer F = V * F;
   Hacer J = V * J.
4. Retorno a lsqcurvefit

Para la realización del ciclo del paso 2 se utilizó la función "eval", para lograr ejecución de instrucciones en MATLAB que puedan modificarse dinámicamente.

Justifiquemos las dos últimas asignaciones de Fun (las del paso 3). La primera corresponde al segundo término de (3.3). Para justificar la segunda asignación, probemos que

$$J_{V*F} = V * J_F$$

donde J_{V*F} y J_F son las matrices jacobianas de $V*F$ y F, respectivamente.

En efecto, de acuerdo con (1.5) sea $Q = (N+1)q$; *a* el rango de la matriz *V* original, o sea, el contenido de rango_V, y m la cantidad de parámetros. Sea además:

$$F = \left(F_k\right)_{k=1,\ldots,Q} \quad ; \qquad V = \left(v_{ik}\right)_{\substack{i=1,\ldots,a \\ k=1,\ldots,Q}}$$

Entonces

$$V \cdot F = \left(v_{i\bullet} \cdot F\right)_{i=1,\ldots,a}$$

También, de acuerdo con (1.4):

$$J_{V*F} = \left(\frac{\partial}{\partial c_j}\left(v_{i\bullet} \cdot F\right)\right)_{\substack{i=1,\ldots,a \\ j=1,\ldots,m}}$$

$$= \left(\frac{\partial}{\partial c_j}\sum_{k=1}^{Q} v_{ik} \cdot F_k\right)_{\substack{i=1,\ldots,a \\ j=1,\ldots,m}}$$

$$= \left(\sum_{k=1}^{Q} v_{ik} \cdot \frac{\partial}{\partial c_j} F_k\right)_{\substack{i=1,\ldots,a \\ j=1,\ldots,m}}$$

$$= V \cdot J_F \qquad \blacksquare$$

Nota: En caso de que este programa vaya a suministrar matriz jacobiana, la función lsqcurvefit exige que se brinde la de F, que difiere en el signo de la matriz jacobiana de r definida formalmente en (1.4).

4. ESTIMACION DE LA MATRIZ DE VARIANZA Y COVARIANZA DIFICULTADES PARA CALCULAR V

Como sabemos, en MCG_Aju_dat_fun_vec, uno de los parámetros de entrada es V, la matriz de varianza y covarianza, cuyo orden es $(N+1)q$, por lo que es probable que a un usuario le resulte difícil o trabajoso suministrar dicha matriz, aunque sea aproximadamente. Es por ello que se presentará un algoritmo para estimarla.[13]

Para el cálculo de V utilizando la conocida fórmula

$$Cov(Y,X) = \frac{\sum_{i=1}^{n}(y_i-\bar{y})(x_i-\bar{x})}{n-1} \qquad (4.1)$$

[13] Como se verá posteriormente en el ejemplo, a pesar de las dimensiones de V, su información útil es a menudo de un orden mucho menor.

necesitamos tener $(N+1)q$ vectores, lo que –suponiendo que cada vector tenga n componentes– conlleva conocer $n(N+1)q$ números, donde *n* debe ser algo grande, de manera que la estimación calculada sea suficientemente buena. Sin embargo, en "Observaciones" sólo disponemos de $(N+1)q$ números, que es mucho menos.

Si nos atenemos estrictamente a la información con que contamos, para cada individuo y cada característica tenemos solamente un vector trivial, que es simplemente un escalar. De esta manera se individualizaría totalmente la información en cada una de las características, pero por supuesto, el cálculo de las covarianzas carecería totalmente de confiabilidad.

Yendo al otro extremo, si utilizamos (2.1), tendríamos muestras suficientemente grandes, pero todas las características quedarían globalizadas, al no reconocerse las posibles diferencias entre individuos.

¿Que hacer entonces?
Ante la imposibilidad de generar una V individualizada completamente (por no contar con suficiente información) adoptaremos una estrategia de compromiso entre ambas posiciones extremas.

Para ello, se ordena previamente la información de acuerdo con los individuos, y posteriormente se hace una partición de las observaciones en grupos (o "clusters"), de manera que todos los individuos de un mismo grupo (es decir, aquellos cuyas observaciones están contenidas en un mismo grupo) sean relativamente cercanos o afines. Entonces, para el cálculo de la covarianza correspondiente a dos características específicas de un par dado de individuos, tomaremos como muestras los grupos en los que se encuentran, y esta covarianza será la misma para todos los individuos de dichos grupos.

Características generales del algoritmo
El algoritmo que lleva a cabo la idea bosquejada en el párrafo anterior, también fue implementado en MATLAB 7.3, en esta ocasión mediante la función "Estimar_mat_Cov". Este subprograma le brinda al usuario diferentes opciones, la primera de ellas es decidir el nivel de detalle para la estimación de V:

1. 'Individuos no correlacionados': Corresponde con (2.1), en este caso no se forman grupos.
2. 'Grupos no correlacionados': Cada grupo tiene covarianzas independientes, pero no hay correlación entre individuos de grupos distintos.
3. 'Grupos correlacionados': Cada grupo tiene covarianzas independientes y en general, hay correlación entre individuos de grupos distintos.

En los casos 2 y 3, se puede imponer la cantidad mínima de observaciones que se desea que haya en cada grupo, y así alcanzar mayor precisión al calcular (4.1). También puede exigirse que todos los grupos tengan el mismo tamaño; esto tiene ventajas y desventajas, como veremos después.

El espacio de las unidades experimentales o individuos puede tener una sola dimensión (como en el caso de unidades de tiempo) o varias dimensiones, cuando por ejemplo se trate de puntos del plano o del espacio tridimensional. En general, por convención, las columnas correspondientes a los individuos van a ser las primeras columnas de Valores_var_indep; por lo tanto es posible que un mismo individuo aparezca repetido, de manera que le correspondan varias observaciones.

Si no se le pide al algoritmo que todos los grupos tengan el mismo tamaño, este tratará, como es natural, que todas las observaciones correspondientes a un mismo individuo estén incluidas en un mismo grupo. Es por esa razón que en este caso el subprograma le pregunta al usuario cuántas dimensiones tiene el espacio de los individuos (dim_indiv). Entonces, los individuos están determinados por las primeras dim_indiv columnas de Valores_var_indep.

Si denotamos mediante G_I el grupo *I*-ésimo, y *C* la cantidad de grupos, entonces podemos decir que G_I es una matriz de n_I filas y *q* columnas, $I = 1, \dots, C$.

Cuando hablamos de "tamaño" del grupo *I*-ésimo, nos referimos al valor de n_I .

Vimos en A1 que, para formar la matriz V, consideramos la partición

$$V = \left(V_{LH}\right)_{\substack{L=1,\ldots,q \\ H=1,\ldots,q}} = \begin{pmatrix} V_{11} & \cdots & V_{1H} & \cdots & V_{1q} \\ \vdots & & \vdots & & \vdots \\ V_{L1} & \cdots & V_{LH} & \cdots & V_{Lq} \\ \vdots & & \vdots & & \vdots \\ V_{q1} & \cdots & V_{qH} & \cdots & V_{qq} \end{pmatrix}$$

donde cada V_{LH} es un bloque cuadrado de orden $N+1$

Si al llamar a Estimar_mat_Cov seleccionamos para la estimación de V la opción 'Grupos correlacionados', el bloque genérico V_{LH} de la matriz resultante quedará a su vez particionado en los sub-bloques $S_{IJ}; I, J = 1, \ldots, C$, y tendrá la estructura siguiente:

S_{11}	S_{12}	…	S_{1J}	…	S_{1C}
S_{21}	S_{22}	…	S_{2J}	…	S_{2C}
⋮	⋮		⋮		⋮
S_{I1}	S_{I2}	…	S_{IJ}	…	S_{IC}
⋮	⋮		⋮		⋮
S_{C1}	S_{C2}	…	S_{CJ}	…	S_{CC}

Estos C^2 sub-bloques en general no son del mismo tamaño.

Sea

$$m = min\{n_I, n_J\}$$

Entonces

$$S_{IJ} = Cov\left(G_I(1 \ldots m, L), G_J(1 \ldots m, H)\right) \cdot \begin{pmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{pmatrix}_{n_I \times n_J} \tag{4.2}$$

Como puede apreciarse, los sub-bloques de la diagonal son cuadrados.

En el caso particular que todos los grupos tengan el mismo tamaño m, es posible dar una expresión más directa para V. Sea

$$G = [G_1(\bullet,1) \ldots G_C(\bullet,1) G_1(\bullet,2) \ldots G_C(\bullet,2) \ldots G_1(\bullet,q) \ldots G_C(\bullet,q)]$$

(operación de concatenación)

y

$$A = \left(a_{ij}\right)_{Cq \times Cq} = Cov(G) \tag{4.3}$$

Entonces

$$V = A \otimes \begin{pmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{pmatrix}_{m \times m} \tag{4.4}$$

Ahora bien, si al llamar a Estimar_mat_Cov seleccionamos para la estimación de V la opción 'Grupos no correlacionados', en el bloque genérico V_{LH} de la matriz resultante,
S_{IJ} quedará definido así:

$$S_{IJ} = \begin{cases} Cov\big(G_I(\bullet, L), G_I(\bullet, H)\big) \cdot \begin{pmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{pmatrix}_{n_I \times n_I} & si\ I = J \\ 0_{n_I \times n_J} & si\ I \neq J \end{cases} \tag{4.5}$$

Observemos que la definición anterior coincide con (4.2) cuando $I = J$, y que ahora $V_{LH}\ es$ diagonal por bloques.

En particular, si todos los grupos tienen el mismo tamaño m , definimos a partir de A en (4.3), la matriz $B = \left(b_{ij}\right)_{Cq \times Cq}$:

$$b_{ij} = \begin{cases} a_{ij} & si\ i \equiv j(mod\ C) \\ 0 & en\ otro\ caso \end{cases} \quad i, j = 1, \ldots, Cq$$

y entonces

$$V = B \otimes \begin{pmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{pmatrix}_{m \times m}$$ ■

Pasos esenciales del subprograma

Los pasos principales de Estimar_mat_Cov son los siguientes:

1. Asignación de los parámetros de entrada: Valores_var_indep y Observaciones
2. Asignar a p el número de columnas de Valores_var_indep, y a q el número de columnas de Observaciones (de acuerdo con (1.1)).
3. Hacer Info=[Valores_var_indep Observaciones][14] (operación de concatenación).
4. Ordenación ascendente de Info
5. Asignar a Val_var_indep_ord las primeras p columnas de Info, y a Observ_ord las últimas q columnas de Info.
   (Estos son parámetros de salida de Estimar_mat_Cov**)**
6. Asignar a cant_obs (cantidad de observaciones) el número de filas de Valores_var_indep.
7. Preguntarle al usuario con qué nivel de detalle desea estimar V, y registrar la respuesta en nivel_detalle.[15]
8. En el caso de que nivel_detalle corresponda con 'Individuos no correlacionados':
   8.1 Calcular V de acuerdo con la igualdad (2.1).
   8.2 Retornar (al programa que invocó a Estimar_mat_Cov).
9. Preguntar cuál será la cantidad mínima de observaciones en cada grupo, y asignarla a cant_min.
10. Si cant_obs<cant_min (la cantidad de observaciones es insuficiente):
    10.1Calcular V de acuerdo con (2.1).
    10.2Señalar la insuficiencia mediante un mensaje.
    10.3Retornar.
11. Preguntar si se desea que todos los grupos tengan el mismo tamaño.
12. Caso de 'Sí':
    12.1 Hacer $cant\_fil\_g = min\{j | cant\_min \leq j \leq cant\_obs \wedge {}^{j}|cant\_obs\}$.
    (Cantidad de filas que tendrá cada grupo)
    12.2 Asignar a Cant_gru la cantidad de grupos.
    12.3 Para cont_grup[16] desde 1 hasta Cant_gru, repetir:
    Formar Grupos{cont_grup}.
    12.4 Hacer cant_fil_gru(i)[17]= cant_fil_g, para todo i=1,…, Cant_gru
    (pues todos los grupos tienen la misma cantidad de observaciones).
13. Caso de 'No':

[14] Información inicial
[15] Esa y otras preguntas se realizan mediante menús o mediante ventanas de diálogo, algunas de las cuales serán vistas en el § 5.
[16] contador de grupos
[17] cantidad de filas del grupo i

13.1 Preguntar cuántas dimensiones tiene el espacio al que pertenecen los individuos, y asignarlo a dim_indiv.
13.2Hacer cont_grup=0 (contador de grupos).
13.3Hacer i=1 (i controla el ciclo de las observaciones).
13.4Mientras sea i<=cant_obs repetir: (se seguirán añadiendo observaciones)
13.4.1 Hacer k=0 (contador interno dentro de un grupo).
13.4.2 Iniciar gru como matriz vacía.
13.4.3 Mientras sea k<cant_min y no esté vacia Info, repetir:
13.4.3.1 De acuerdo con el valor de dim_indiv, seleccionar *todas* las filas de Info correspondientes al primer individuo, y guardar esta matriz en Info_indiv_1.
13.4.3.2 Eliminar las primeras p columnas de Info_indiv_1. (Ahora queda sólo la información básica del individuo.)
13.4.3.3 Añadir todas las filas de Info_indiv_1 a las filas de gru.
13.4.3.4 Asignar a cant_indiv_1 la cantidad de veces que aparece el individuo 1 (es igual a la cantidad de filas de Info_indiv_1).
13.4.3.5 Hacer k=k+ cant_indiv_1; i=i+ cant_indiv_1.
13.4.3.6 Eliminar de Info las filas correspondientes a Info_indiv_1.
13.4.4 Incrementar cont_grup.
13.4.5 Hacer Grupos{cont_grup} = gru.
13.4.6 Guardar en cant_fil_gru(cont_grup) el numero de filas de gru.
13.5 Hacer Cant_gru = cont_grup. (Cantidad de grupos)
13.6 Si cantidad de filas de Grupos{Cant_gru}<cant_min:[18]
13.6.1 Añadir todas las filas de Grupos{Cant_gru} a Grupos{Cant_gru-1}.
13.6.2 Actualizar cant_fil_gru(Cant_gru-1) y decrementar Cant_gru.
14. Si Cant_gru = 1, advertir que hay un solo grupo de observaciones.
15. Si nivel_detalle corresponde con 'Grupos no correlacionados':
15.1Para cont_grup desde 1 hasta Cant_gru, repetir:
Calcular cov([Grupos{cont_grup} Grupos{cont_grup}]), e ir formando V de acuerdo con (4.5).
15.2Retornar

Finalmente, caso en que hay correlación entre individuos de grupos distintos:

16. Para cont_grup desde 1 hasta Cant_gru, repetir:
Para j desde 1 hasta Cant_gru, repetir:
Calcular covarianzas entre los grupos cont_grup y j, e ir formando V de acuerdo con (4.2).
17. Retornar.

INFLUENCIA DE LA SELECCIÓN DE ALGUNAS OPCIONES EN LA ESTIMACION DE V

Zoonekynd [2007] señala que, en la estimación de la matriz de correlación, la no utilización del mismo número de observaciones (tamaño de muestra) para calcular todos los coeficientes, trae como consecuencia que la matriz de correlación no necesariamente tiene que ser definida positiva.

Como consecuencia, en este caso V tampoco tiene que serlo. Esto constituye una desventaja de la opción de no exigir que todos los grupos tengan igual tamaño al invocar a Estimar_mat_Cov.
Por otra parte, la opción contraria de formar todos los grupos con igual tamaño, tiene las siguientes desventajas:

1. Es posible que toda la información correspondiente a algún individuo no esté incluida en un mismo grupo.[19]
2. Si el único entero mayor o igual que cant_min y divisor de cant_obs es el propio cant_obs, entonces habrá un solo grupo, y no se individualizará la información (ver instrucción 12.1 del algoritmo).[20]

[18] entonces el último grupo es demasiado pequeño.
[19] De cualquier manera esto no podría ocurrir si a cada individuo le corresponde una sola observación.
[20] En particular esto ocurre cuando cant_obs es primo.

En general, Estimar_mat_Cov produce siempre una matriz simétrica, pero como consecuencia de que los grupos no sean del mismo tamaño o de que estos no sean suficientemente grandes para el cálculo de (4.1), es posible que aparezca algún valor propio pequeño de V que sea negativo, lo que atentaría contra el hecho de que esta debe ser definida no negativa. Si esto ocurre, el usuario puede:

1. Aumentar el valor de cant_min.
2. Utilizar otra opción para la generación de V.
3. No hacer nada, e introducir la misma V para hacer el ajuste de datos. En este caso, MCG_Aju_dat_fun_vec filtrará el valor propio negativo y el vector propio correspondiente.

Demostraremos a continuación que, si se aplica la opción 3, con el filtraje no se introduce un error adicional significativo en la estimación de V:

Sea V una matriz simétrica de orden Q. Sabemos que, por la descomposición espectral:

$$V = U\Lambda U^T \tag{4.6}$$

donde $\Lambda = diag(\lambda_1, \lambda_2, \dots, \lambda_Q)$; $U = (v_1 v_2 \cdots v_Q)$ y las columnas $(v_j)_{j=1,\dots,Q}$ son los vectores propios correspondientes a (λ_j), formando una base ortonormal en $\Re^Q$. Sin pérdida de generalidad, podemos suponer que $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_Q$.

Pero

$$\Lambda = \begin{pmatrix} \Lambda_- & 0 \\ 0 & \lambda_Q \end{pmatrix}$$

y

$$U = (U_- v_Q)$$

Sustituyendo en (4.6):

$$V = (U_-\Lambda_- \quad \lambda_Q v_Q)\begin{pmatrix} U_-^T \\ v_Q^T \end{pmatrix}$$

$$= U_-\Lambda_- U_-^T + \lambda_Q v_Q v_Q^T$$

$$V = \overline{V} + \lambda_Q v_Q v_Q^T$$

$$\left\|V - \overline{V}\right\|_2 \leq |\lambda_Q| \|v_Q\|_2 \|v_Q^T\|_2 = |\lambda_Q|$$

por ser normales los v_j . Entonces el error absoluto adicional cometido es a lo sumo el valor absoluto del valor propio despreciado. Con respecto al error relativo, se tiene que

$$\frac{\left\|V - \overline{V}\right\|_2}{\|V\|_2} \leq \frac{|\lambda_Q|}{|\lambda_1|}$$

ya que, como V es simétrica, se cumple que $\sigma_j = |\lambda_j|$, donde σ_j es un valor singular de V, para $j = 1, \dots, Q$.

En la desigualdad anterior, como λ_1 es el valor propio de mayor magnitud, la cota es un número relativamente pequeño, dentro de nuestro problema. ■

5. EJEMPLO

Sea

$$f : \Re^2 \to \Re^2 \; ; \quad f(u, y) = (\operatorname{sen}(u + y), \ln(2 + u - y))$$

Como valores de la variable independiente, consideremos todos los pares $(u, y) : u = 0.2i; y = 0.2j;\ i, j = 0, \dots, 5.$ De manera más precisa (de acuerdo con la notación utilizada en el §1 para definir AD_MCG_FV):

Tomando en cuenta que $k = 0, \dots, 35$ puede escribirse de manera única como $k = 6b + d;\ b, d = 0, \dots, 5$, entonces $x_k = (0.2b, 0.2d)$.

Como observaciones, se tomaron las imágenes exactas de los 36 pares (u, y) por *f*, y cada una fue alternadamente disminuida o aumentada en un 5%; o dicho más formalmente

$$f_k = f(x_k) - (-1)^k \times 0.05 f(x_k)$$

Consideremos ahora

$$F(u, y; c_0, c_1) = \exp(c_0 + u + y) \ln(2 + c_1 u - y) \tag{5.1}$$

Se quiere realizar AD_MCG_FV. Por supuesto, al resolver este problema debe obtenerse *aproximadamente* $c_0 = 0;\ c_1 = 1$.

Correlacion existente

Si ahora escribimos ("reshape") Observaciones como una matriz 12×6, obtenemos:

0	0.3699	0.6815	0.6585	0.8317	0.9781
0.2086	0.5929	0.8835	0.6172	0.8279	1.0033
0.3699	0.6815	0.8854	0.4465	0.6585	0.8317
0.5929	0.8835	1.0347	0.3533	0.6172	0.8279
0.6815	0.8854	0.9496	0.1732	0.4465	0.6585
0.8835	1.0347	1.0225	0	0.3533	0.6172
0.1887	0.5364	0.7994	0.7490	0.9077	1.0437
0.4089	0.7532	0.9786	0.7278	0.9192	1.0811
0.5364	0.7994	0.9362	0.5584	0.7490	0.9077
0.7532	0.9786	1.0496	0.4935	0.7278	0.9192
0.7994	0.9362	0.9252	0.3196	0.5584	0.7490
0.9786	1.0496	0.9548	0.1914	0.4935	0.7278

(5.2)

Si dividimos el conjunto de individuos en tres grupos de 12 individuos cada uno, podemos considerar entonces que en la matriz anterior, las tres primeras columnas corresponden a la primera característica de los tres grupos respectivamente; y similarmente las últimas tres columnas con respecto a la segunda característica.

La matriz de coeficientes de correlación es:

1.0000	0.9858	0.7775	-0.8128	-0.7945	-0.7631
0.9858	1.0000	0.8722	-0.7718	-0.7457	-0.7054
0.7775	0.8722	1.0000	-0.5218	-0.4773	-0.4183
-0.8128	-0.7718	-0.5218	1.0000	0.9955	0.9799
-0.7945	-0.7457	-0.4773	0.9955	1.0000	0.9944
-0.7631	-0.7054	-0.4183	0.9799	0.9944	1.0000

De acuerdo con el penúltimo párrafo, si dividimos la matriz anterior en cuatro bloques cuadrados $(B_{LH})_{\substack{L=1,2 \\ H=1,2}}$ tenemos en cada bloque B_{LH} los coeficientes de correlación correspondientes a las características *L* y *H* de los tres grupos.

Puede observarse entonces la alta correlación existente entre las características de los distintos grupos.[21] Esto hace que la aplicación de los modelos que se están presentando aquí sea adecuada.

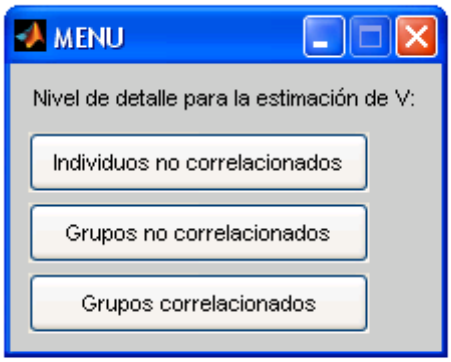
Por otra parte, las covarianzas correspondientes a (5.2) son:

[21] Recordemos que, a diferencia de $Cov(Y, X)$, $Cor(Y, X)$ es invariante por cambios en la unidad de medida, y siempre toma valores en $[-1,1]$.

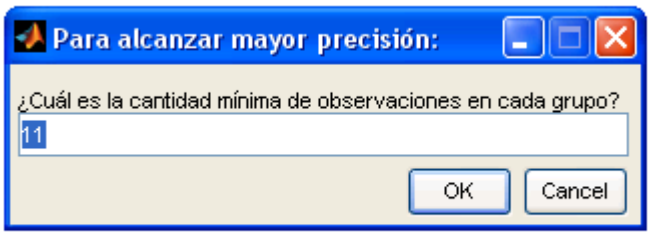
$$\begin{matrix} 0.0924 & 0.0636 & 0.0247 & -0.0588 & -0.0446 & -0.0353 & \\ 0.0636 & 0.0450 & 0.0193 & -0.0390 & -0.0292 & -0.0227 & \\ & 0.0247 & 0.0193 & 0.0109 & -0.0130 & -0.0092 & -0.0066 \\ -0.0588 & -0.0390 & -0.0130 & 0.0566 & 0.0438 & 0.0354 & \\ -0.0446 & -0.0292 & -0.0092 & 0.0438 & 0.0342 & 0.0279 & \\ -0.0353 & -0.0227 & -0.0066 & 0.0354 & 0.0279 & 0.0231 & \end{matrix} \quad (5.3)$$

Estimación de V

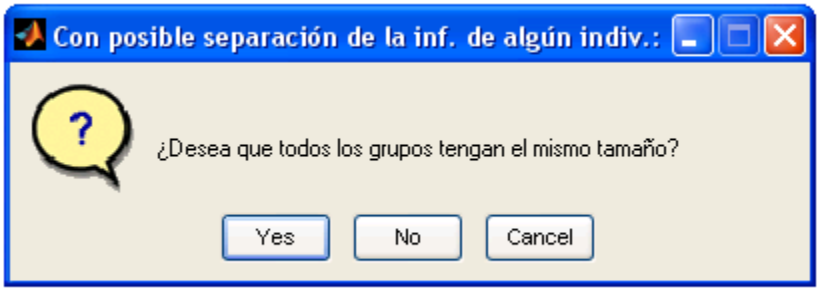
Para ejecutar el software realizado comencemos con la estimación de V mediante Estimar_mat_Cov. Inicialmente, aparece el



Suponga que seleccionamos "Grupos correlacionados". A continuación se abre la ventana



Aceptemos la cantidad por defecto. La siguiente pregunta es:



Supongamos que la respuesta es afirmativa (por ejemplo), entonces –de acuerdo con el paso 12.1 del algoritmo– se formarán 3 grupos de 12 filas cada uno.

Por la instrucción 5 sabemos que los valores ordenados de la variable independiente y las observaciones correspondientes son parámetros de salida.[22]

Comentemos sobre el otro parámetro de salida, es decir, V. Como se seleccionó la opción de igual tamaño para todos los grupos, podremos ver con claridad la estimación de $V_{72\times72}$ hecha por el subprograma, mediante la sustitución directa en (4.4), haciendo A igual a la matriz (5.3) y $m = 12$.

Verifiquemos además la simetría y la definición no negativa de V. En efecto:
>> norm(V-V')[23]

ans =

1.8642e-016

>> min(real(eig(V)))[24]

ans =

-1.1832e-016

Dentro de la aritmética de punto flotante, estos valores son despreciables.

Ajuste de datos
Una vez estimada V, pasemos ahora a ejecutar MCG_Aju_dat_fun_vec. Toda la información de entrada es la siguiente: Valores de la variable independiente y las observaciones correspondientes[25], aproximación inicial, que se tomó como $(.5 \quad .5)^T$, y la V, cuya estimación se obtuvo con anterioridad. Además se suministraron las componentes de la función vectorial (5.1) y sus matrices jacobianas, es decir F1, J1, F2 y J2.

Al ejecutarse el programa se obtuvo la siguiente información de salida (ver MathWorks [2006]):
Optimization terminated: relative function value
changing by less than OPTIONS.TolFun.
vec_param =
0.0010
0.9999
norma_resid_al_cuad =
2.0891e-004
residual =
-0.0124
0.0063
0.0010
-0.0007
0.0038
indicador =
3 [26]
rango_V =
5
otras_informac =

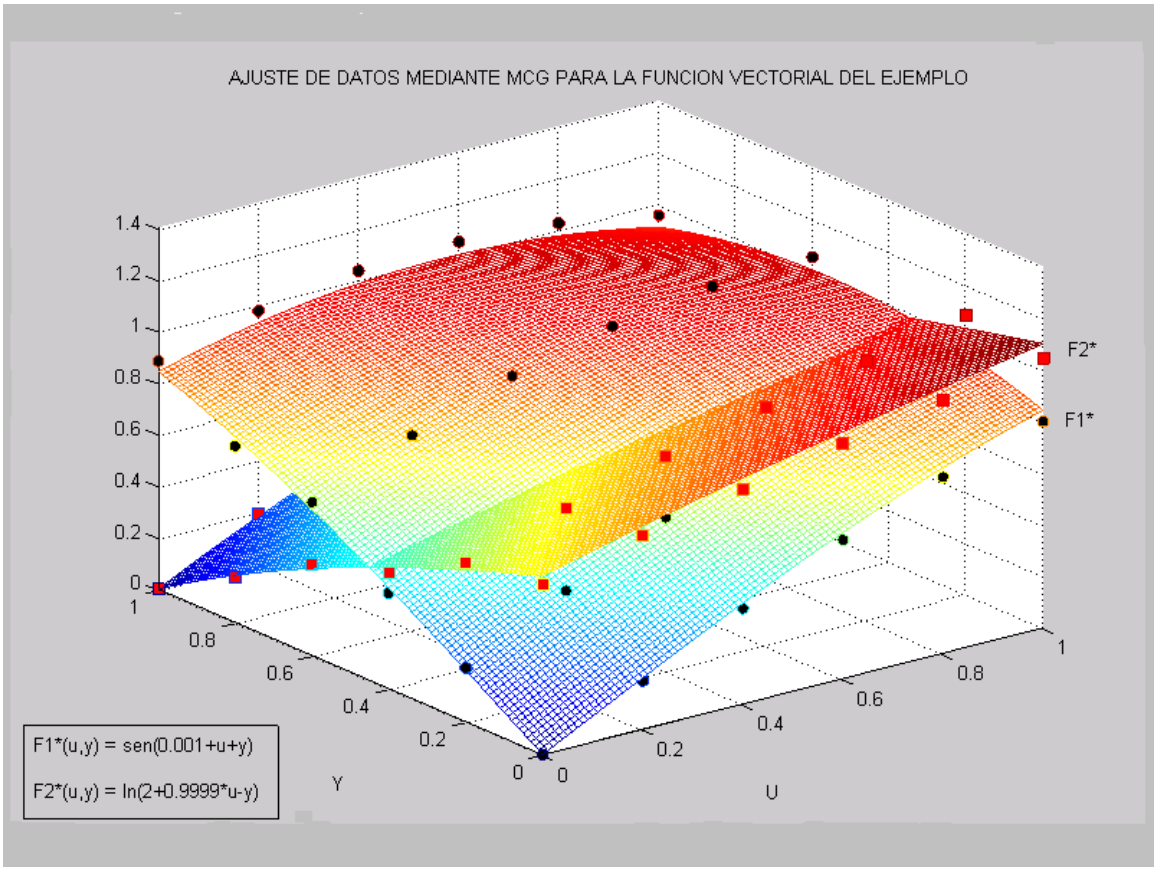
[22] En el caso de este ejemplo en particular, no hubo modificación con respecto a los parámetros de entrada.
[23] Se refiere a $\| \quad \|_2$.
[24] Aunque en teoría los valores propios en este caso deben ser reales, es posible que, producto de las aproximaciones, aparezcan componentes imaginarias pequeñas. De cualquier manera, posteriormente la función "MCG_Aju_dat_fun_vec" las filtrará.
[25] En nuestro caso ya fueron obtenidos anteriormente de manera ordenada a partir de Estimar_mat_Cov.
[26] Indica que el cambio en el residual fue menor que la tolerancia especificada.

firstorderopt: 3.4512e-006
iterations: 4
funcCount: 5
cgiterations: 4
algorithm: 'large-scale: trust-region reflective Newton'
message: [1x87 char]

Estos resultados concuerdan con lo que esperábamos. La aproximación óptima

$$F^*(u,y) = \left(F_1^*(u,y),\ F_2^*(u,y)\right)$$

que hemos obtenido, así como los conjuntos discretos de puntos están representados en el siguiente gráfico:



Debemos señalar que al resolver este mismo problema mediante MCO, tomando como referencia $c^* = (0 \quad 1)^T$, el error en vec_param es unas 3.4 veces mayor que el que obtuvimos anteriormente, lo que también resulta normal (ver final del §1). ■

6. CONCLUSIONES

1. De acuerdo con los objetivos propuestos, se ha presentado aquí un algoritmo implementado para realizar AD_MCG_FV, donde la matriz V de varianza y covarianza (correspondiente a la vectorización de la matriz de observaciones) es simétrica y definida no negativa (no necesariamente definida positiva), y las componentes de la función vectorial son en general no lineales.
2. También se ha presentado un algoritmo implementado para estimar dicha V, si el usuario lo desea, con diferentes opciones.

3. En general –en aras de la eficiencia– los subprogramas elaborados explotan las facilidades de MATLAB para las operaciones matriciales, y se toma en cuenta que el orden de V puede ser grande.

RECEIVED NOVEMBER 2008
REVISED SEPTEMBER 2010

REFERENCIAS

[1] ALZOLA, C. and HARRELL, F. (2002): **An Introduction to S and the Hmisc and Design Libraries**, http://hesweb1.med.virginia.edu/biostat/s/doc/splus.pdf

[2] CHATTERJEE, S. and HADI, A. S. (2006): **Regression analysis by example, 4th edition**, Wiley series in Probability and Statistics, Hoboken, New Jersey.

[3] CUÉ, J. L., HERNÁNDEZ, N. y CASTELL, E. (1985): **Modelo Lineal y sus aplicaciones**, Editorial Pueblo y Educación, La Habana

[4] GALLANT, A. R. (2000): **Nonlinear Statistical Models, Chapter 5, Multivariate Nonlinear Regression**, http://www.unc.edu/~arg/econ275/lectures/ch5sld4.ps

[1] GILL, P. E., W. MURRAY and M. H. WRIGHT (1990): **Numerical Linear Algebra and Optimization, Volume 1,** Addison-Wesley

[1] HAGER, W. W. (1988): **Applied Numerical Linear Algebra**, Prentice Hall

[5] HENGL, T. (2007): **A Practical Guide to Geostatistical Mapping of Environmental Variables**, http://eusoils.jrc.it/ESDB_Archive/eusoils_docs/other/EUR22904en.pdf

[6] HINTZE, J. L. (2007): **NCSS User's Guide III, Regression and Curve Fitting**, http://itchy.icw.com/ncss/NCSSUG3.pdf

[7] HYAMS, D. (2003): **CurveExpert 1.3,** http://s91928265.onlinehome.us/curveexpert/downloads/cxptw138.zip

[8] LOHNINGER, H. (2008): **DataLab, Version 2.599, Epina GmbH,** http://www.lohninger.com/datalab/en_download.html

[9] MARANGONI, F. (1999): **Covariance matrix of a multivariate non linear regression model**, http://www.eco.rug.nl/gauss/gauss99/msg00783.html

[10] MATHWORKS INC., THE (2004): **Model-Based Calibration Toolbox, Model Browser User's Guide**, *Version 2*, Natick, MA, http://biounder.kaist.ac.kr/board/bx/docs/matlabman/mbcmodel.pdf

[11] MATHWORKS INC., THE (2006): **MATLAB Version 7.3.0.267**, http://www.mathworks.com/access/helpdesk/help/techdoc/matlab.shtml

[12] MEYER, K. (2007): **WOMBAT – A tool for mixed model analyses in quantitative genetics by restricted maximum likelihood (REML***)*, http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2064953

[13] MULTIRESPONSE PARAMETER ESTIMATION (actualizado 2003), Chapter 4, Statistics 824 – Spring 1995, http://www.stat.wisc.edu/p/stat/course/st824-bates/public/slides/NRAIA4_sem.ps

[14] R (2008): **The Comprehensive R Archive Network**, http://cran.r-project.org/doc/

[15] SAHA, S. (2003): **Designs for Multiresponse models**, http://www.stat.ufl.edu/~ssaha/AdvancedDesignReport.pdf

[16] SAS (2008): **SAS/STAT® 9.2 User's Guide**, http://support.sas.com

[17] SCIENTIFIC SOFTWARE INTERNATIONAL, INC (2003): **Lisrel,** http://www.ssicentral.com/lisrel/mainlis.htm

[18] SHERROD, P. H. (2003): **NLREG, Nonlinear Regression Analysis Program**, http://www.nlreg.com/NLREG.pdf

[19] SMITH, D. L. (1993): **A Least-Squares Computational "Tool Kit",** http://www.osti.gov/energycitations/servlets/purl/10176638-uUIhTq/10176638.PDF

[20] STRUCTURAL EQUATION MODELING (2002): http://www2.uta.edu/sswmindel/S6341/Class Lecture Sup/SEM/Principles of SEM.pdf

[21] UCLA(no aparece fecha): **Stat Consulting Support Policies for R**, UCLA Academic Technology Services, USA, http://www.ats.uda.edu/stat/r/StatConsultingSupportR.htm

[22] ZANELLA, A., BOARI, G. and ZAPPA, D. (2000): **Multivariate Models and Methods in Technology**, Istituto di Statistica, Università Cattolica del Sacro Cuore, Milano, htttp://www.dms.unina.it/sis2003/Lavori/Zanella.pdf

[23] ZOONEKYND, V. (2007): **Regression**, http://zoonek2.free.fr/UNIX/48_R/09.html