STOCHASTICA Vol. IX Nº 3, 1985.

UN ALGORITMO PARA CONSTRUIR CODIFICACIONES DIFUSAS BALANCEADAS

Glz. de Garibay V. y Barba L.

ABSTRACT

Results assuring the existence of semilinear fuzzy partitions in 2,3 or 4 balanced fuzzy classes are known. Such existence is not guaranteed for a greater number of classes.

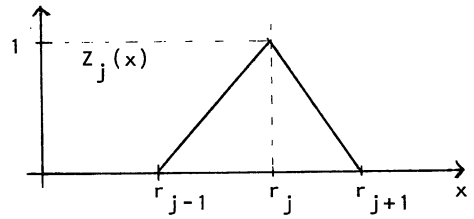
In this paper we present an algorithm that characterizes the set of solutions, and constructs any of them. Situations without a solution are also detected.

We give a FORTRAN program for the algorithm and some examples.

1. Introducción

Diversos autores vienen utillzando codificaciones difusas como alternativa a la codificación disyuntiva, que presenta problemas de pérdida de información (a partir de los valores codificados no pueden struirse los datos originales) y de discontinuidad (aparecen distancias artificiales entre puntos próximos que al ser codificados pertenecen a clases diferentes). El uso de c.d. se impone en determinadas circunstancias (Roux y Guitonneau 1977, Lefol 1979, Gallego 1980 y 1982, Cazes 1980, Garibay 1984 y 1985).

Una c.s.l.d. (c.d. semilineal) (Gallego 1982) en J clases difusas está asociada a una partición difusa de la Recta (Dubois 1980) cuyasf.p. (funciones de pertenencia) $Z_1,\ldots Z_J$ están determinadas por J valores reales $r_1<\ldots<r_J$ llamados p.r. (puntos de referencia), de la siguiente manera:



$Z_j(x) = I_j(x).\ (x-r_{j-1})/(r_j-r_{j-1})+I_{j+1}(x).(r_{j+1}-x)/(r_{j+1}-r_j)$, con las lógicas modificaciones $Z_1(x)=1$ si $x<r_1$ y $Z_J(x)=1$ si $x>r_J$. (Con I_j representamos el indicador del intervalo $[r_{j-1},r_j)$).

Consideraremos en todo el trabajo la muestra ordenada $x_1<x_2<\ldots<x_n$. Utilizando la notación de Garibay y Barba (1985), llamamos contingente difuso de la clase j al valor T_j definido por $\Sigma_{i=1}^{n} Z_j(x_i)$ y vector T de contingentes difusos a $(T_1,\ldots,T_J)$ (notar que $\Sigma_{j=1}^{n} T_j = n$).

Si es el cono abierto de R^J definido por $r_1<\ldots<r_J$, para cada elemento $o = (r_1,\ldots,r_J)'$ de θ, tenemos la c.s.l.d. asociada estos p.r. cuyas f.p. notaremos $Z_1^o \ldots Z_J^o$. Si no hay lugar a equívocos, usaremos Z_j por Z_j^o.

Consideramos T_j como una aplicación de θ en R, que en o toma el valor $T_j(o)= \Sigma_i\ Z_j^o(x_i)$. De forma análoga, T resulta una aplicación de θ en R^J: $T(o)=(T_1(o),\ldots,T_J(o))'$. Si no hay lugar a equívoco sobre el o utilizado, notaremos T_j por $T_j(o)$ y T por T(o).

Nuestro algoritmo encuentra los elementos o de θ tales que $T(o)=c.1_J$ ($1_J=(1,\ldots,1)' \in R^J$) si existen, y en caso contrario, detecta la no existencia de solución (notar que c=n/J).

2. Resultados previos.

Garibay y Barba (1985) estudian el problema de existencia de soluciones. Demuestran que para J=2,3 y 4, el problema siempre tiene solución y dan contraejemplos que demuestran que para J>4 no puede asegurarse que exista.

Prueban asimismo que $T:\theta \dashrightarrow R^J$ es contínua; que $\forall j=2...J$, si $T_1(o)=...=T_{j-1}(o)=c$, y si además

a) $(j-1)c \notin N$, entonces existe una dependencia funcional (determinada por la muestra) entre r_{j-1} y r_j: $f_{j-1,j}(r_{j-1})=r_j$. Esta $f_{j-1,j}(.)$ resulta monótona estrictamente decreciente, contínua, y si $x \uparrow x_s$, $\lim f_{j-1,j}(x) = x_s$, donde $s=[n(j-1)/J]+1$ ($[x]$ representa la parte entera de x); f es además poligonal.

b) $(j-1)c \in N$, entonces la dependencia $f_{j-1,j}(r_{j-1})=r_j$ resulta para valores $r_{j-1} < x_s$ (siendo ahora $s=n(j-1)/J$), con las mismas características de a), salvo que $\lim f_{j-1,j}(x)=x_{s+1}$ cuando $x \uparrow x_s$; demuestran también que $r_{j-1} \in [x_s, x_{s+1})$ si y solo si $r_j \in (x_s, x_{s+1}]$ y en estos casos definen $f_{j-1,j}(r_{j-1})$ como el intervalo $(r_{j-1}, x_{s+1}]$ probando que $r_j \in f_{j-1,j}(r_{j-1})$; $f_{j-1,j}^{-1}(r_j)$ es el intervalo $(x_s, r_j]$ y demuestran que $r_{j-1} \in f_{j-1,j}^{-1}(r_j)$.

Utilizamos estos resultados y notaciones en el presente trabajo.

3. Desarrollo teórico.

<u>Teorema 1.</u> Sean n cualquiera y J<n; c=n/J; Dados $x_1<...<x_n$ y $o \in \theta$ cualesquiera, entonces:

$$T(o)=1_J.c \iff \forall j=2... J, r_j \in f_{j-1,j}(r_{j-1}).$$

Demostración.

$T_1=c \iff r_2 \in f_{1,2}(r_1)$;

para $j>1$:

$$T_1=\ldots=T_{j-1}=c \iff T_1=\ldots=T_{j-2}=c \text{ y } r_j \in f_{j-1,j}(r_{j-1}) \iff$$

$$\iff r_2 \in f_{1,2}(r_1)\ldots r_j \in f_{j-1,j}(r_{j-1})$$

Luego $T(o)=c.1_J \iff r_j \in f_{j-1,j}(r_{j-1})$ $\forall j=2 \ldots J$.

###

Teorema 2. Sean n cualquiera y $J<n$; $c=n/J$; cada muestra $x_1<\ldots<x_n$ determina J pares $(L_j,U_j) \in R^2$ tales que:

$$L_j<U_j \forall j=1\ldots J \iff \exists\, o \in \theta / \; T(o)=1_J.c$$

Además, si $L_j< U_j$ $\forall j=1\ldots J$, entonces;

$\forall j$, $\forall r_j \in (L_j,U_j)$ y $\forall i \in \{1,\ldots,J\}-\{j\}$, $\exists\, r_i \in (L_i,U_i)$ de forma que

$$o=(r_1,\ldots,r_j)' \in \theta \quad \text{y } T(o)=c.1_J.$$

Demostración.

A partir del Teorema 1 y teniendo en cuenta el rango de definición de $f_{j-1,j}(.)$ y Garibay, Barba (1985), resulta:

$T(o)=1_J.c \implies \forall j=1\ldots J \; r_j<U_j(j)$, siendo $U_j(j)=x_{[jc+1]}$ para $j=1\ldots J-1$ y $U_J(J)=\infty$.

Dadas las propiedades de las $f_{j-1,j}$ (Garibay y Barba 1985 Th. 2 y 3), $T(o)=1_J.c \implies L_j(j)<r_j<U_j(j)$ $\forall j=1\ldots J$, siendo $L_j(j)=x_{(j-1)c}$ si $(j-1)c \in N$ y $L_j(j)=x_{[(j-1)c+1]}$ si $(j-1)c \notin N$ cuando $j=2\ldots J$ y $L_1(1)=-\infty$. (Hagamos notar que $U_j(j)=L_{j+1}(j+1)$ si

jc ∉ N).

Bajo la hipótesis $T(o)=c.1_J$, la acotación $r_j<U_j(j)$ obliga a que tanto r_{j-1} como r_{j+1} estén acotados inferiormente por sendos valores, que llamaremos $L_{j-1}(j)$ y $L_{j+1}(j)$ resp. Análogamente, $r_j>L_j(j)$ obliga a que r_{j-1} y r_{j+1} sean menores que sendas cotas superiores que llamamos $U_{j-1}(j)$ y $U_{j+1}(j)$.

Estas cuatro nuevas cotas afectan ahora a r_{j-2} y a r_{j+2}, obteniendo para estos p.r.:

$$L_{j-2}(j) < r_{j-2} < U_{j-2}(j)$$

y $$L_{j+2}(j) < r_{j+2} < U_{j+2}(j).$$

Partiendo de cada r_j y repitiendo el proceso hasta alcanzar en cada caso las cotas para r_1:$L_1(j),U_1(j)$ y para r_J:$L_J(j),U_J(j)$ obtenemos sendas colecciones de J cotas para cada r_j:

$$L_j(i) < r_j < U_j(i) \quad i=1...J.$$

Así pués, $T(o)=1_J.c$ ===> $L_j<r_j<U_j$ $\forall j=1...J$, siendo para cada j $L_j=\max\{L_j(1),...,L_j(J)\}$ y $U_j=\min\{U_j(1),...,U_j(J)\}$.

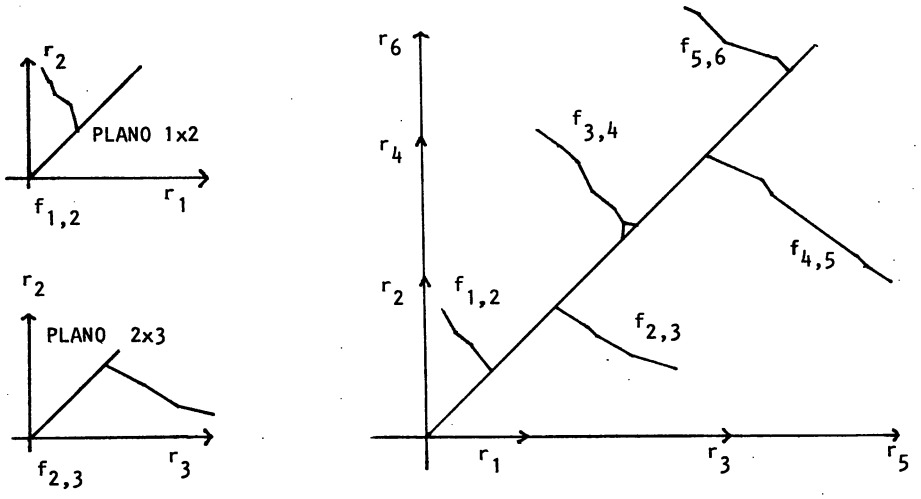
Recíprocamente, si $L_j<U_j$ $\forall j=1...J$, entonces $\forall j$ y $\forall r_j \in (L_j,U_j)$, debido a la construcción de las cotas y a las propiedades de las funciones de dependencia, $f_{j,j+1}(r_j) \cap (L_{j+1},U_{j+1})=A\neq\emptyset$, y puedo elegir un elemento $r_{j+1} \in A$. Análogamente, $f_{j,j-1}^{-1}(r_j)\cap(L_{j-1},U_{j-1})$ es no vacio y tomo r_{j-1} en este conjunto. Reiterando este proce dimiento, el vector $o=(r_1,...,r_j,...,r_J)'$ constituye una solución $o \in \theta$ y $T(o)=c.1_J$.

Por tanto $\exists o \in \theta /\ T(o)=c.1_J$ <===> $L_j<U_j$ $\forall j=1...J$.

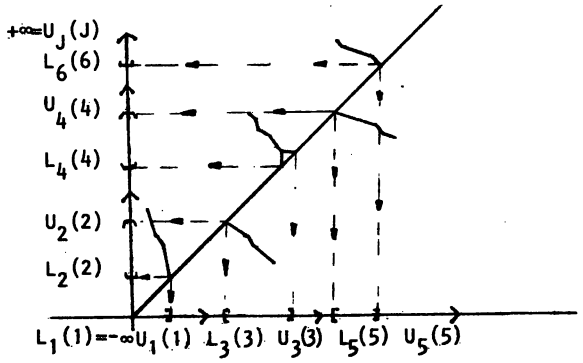
4. El algoritmo.

Los propios pasos de la demostración, constructiva, del Teore ma 2 conforman un algoritmo que (a) si no existe solución, encuen tra $L_i>U_j$ para algún j, y (b) si existe, encuentra los intervalos

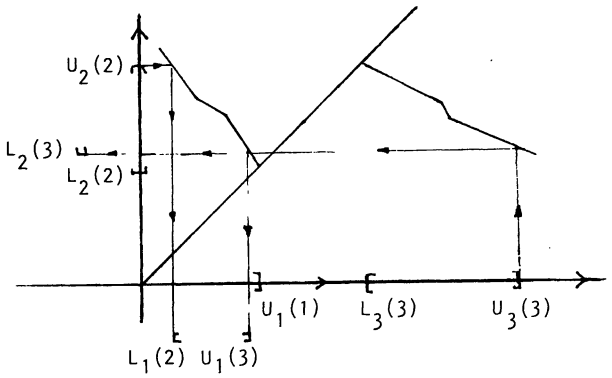
(L_j,U_j) tales que tomando un punto cualquiera de uno cualquiera de ellos, $r_j \in (L_j,U_j)$, podemos completar una J-upla solución, $o=(r_1,\ldots r_J)'$, eligiendo convenientemente los restantes p.r. La idea gráfica del algoritmo es la siguiente: dibujemos el grafo de $f_{j-1,j}$ (que relaciona r_{j-1} y r_j bajo la hipótesis $T(o)=c.1_J$) en el plano (j-1)xj; podemos superponer las J-1 gráficas en una sola, que representa todas las dependencias entre r_{j-1} y r_j bajo la hipótesis de equilibrio:



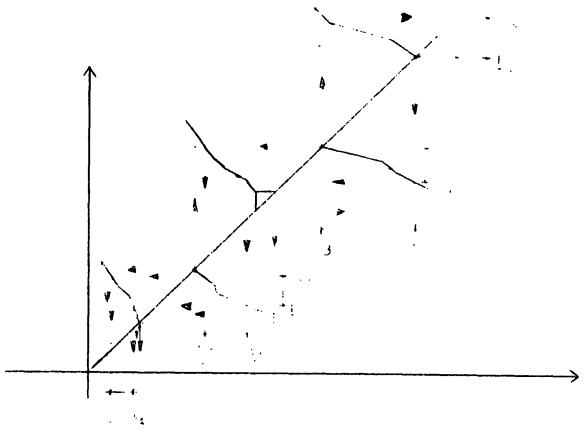
Los valores $L_j(j)$ y $U_j(j)$ (que son los primeros que cargamos en L_j y U_j resp.) se obtienen gráficamente de la forma:



Ilustramos en la siguiente figura la obtención de algunas cotas $U_j(i)$ ó $L_j(i)$ con $i \neq j$:

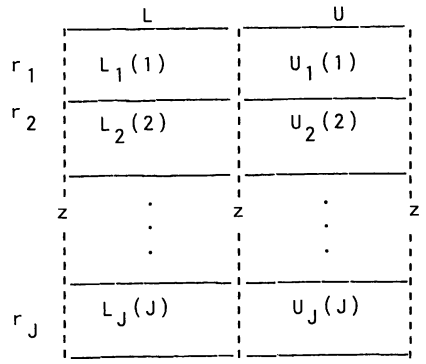


La intersección $\cap_{i=1}^{n} (L_j(i), U_j(i))$ nos da el intervalo solución (L_j, U_j) para r_j. En este ejemplo, cada L_j ha resultado menor que el correspondiente U_j; todos los intervalos (L_j, U_j) son no vacios y tenemos caracterizado el conjunto de todas las soluciones.



Los Pasos del Algoritmo.

Paso 1: Cargamos en L_j el valor $L_j(j)$ y en U_j el $U_j(j)$ $(j=1...J)$:



(Notar que $L_1(1) = -\infty$ y que $U_J(J)=+\infty$).

Definición. Sea L_{j-1} la cota inferior para r_{j-1} almacenada tras

j-1 pasos del algoritmo. En el paso j, tomamos un nuevo p.r., r_j, que está acotado superiormente por U_j. La monotonia de $f_{j-1,j}$ hace que si $r_j<U_j$, entonces $r_{j-1}>a$ para un cierto $\underline{a}$. Diré que $r_j<U_j$ OBLIGA a L_{j-1}, cuando $\underline{a}$ sea mayor que el valor L_{j-1}. En tal caso, a pasa a ser el nuevo valor cargado en L_{j-1}. Análogamente diré que $r_j>L_j$ OBLIGA a U_{j+1} ó a U_{j-1}.

Es inmediato comprobar que si $r_j<U_j$ obliga a L_{j-1}, entonces L_{j-1} NO obliga a U_j, y reciprocamente. (1)

PASO 2: Consideramos r_2

2.1: $r_2>L_2(2)$ nunca obliga a U_1.

2.2: $r_2<U_2(2)$ siempre obliga a L_1. Cargamos en L_1 este valor, mayor que el anterior. Termina Paso 2.

PASO 3: Consideramos r_3

3.1: $r_3>L_3(3)$ nunca obliga a U_2

3.2: Debido a (1), únicamente puede darse una de estas dos situaciones:

3.2.a. $r_2>L_2(2)$ obliga a U_3; cargamos en U_3 este valor, menor que el anterior, y termina el paso 3.

3.2.b. $r_3<U_3(3)$ obliga a L_2; cargamos en L_2 este nuevo valor mayor que el anterior, y revisamos si para este nuevo valor de L_2, $r_2>L_2$ obliga a U_1:

si NO obliga, termina el psaso 3.
si SI obliga, modificamos U_1 y termina el paso 3.

.
.
.

PASO j: Consideramos r_j:

j.1 $r_j > L_j(j)$ nunca obliga a U_{j-1};

si $r_{j-1} < U_{j-1}$ obliga a L_j, cargamos el nuevo valor de L_j.

j.2 Por (1), únicamente puede darse una de estas dos situaciones:

j.2.a. $r_{j-1} > L_{j-1}$ obliga a U_j;

cargamos el nuevo valor de U_j y termina el paso j.

j.2.b. $r_j < U_j(j)$ obliga a L_{j-1};

vemos si este nuevo L_{j-1} obliga a U_{j-2};

si NO obliga, termina el paso j.

si SI obliga, cargo este nuevo U_{j-2}

vemos si este nuevo U_{j-2} obliga a L_{j-3};

si NO obliga, termina el paso j.

si SI obliga, cargo este nuevo L_{j-3}

vemos si este nuevo L_{j-3} obliga a U_{j-4};

... hasta donde sea posible (j-1 veces a lo sumo).

termina paso j.

.
.
.

Continuamos hasta concluir el PASO J.

En este momento, tendremos cargados en $(L_j, U_j)_{j=1...J}$ los valores L_j U_j del Teorema 2.

5. Programa Fortran.

Elaborando un programa en FORTRAN, hemos conseguido la puesta a punto del algoritmo desde un punto de vista práctico y resulta así cómodamente utilizable. En el apéndice 2 incluimos un somero resumen del diagrama de flujo empleado.

Al correr el programa, debemos dar valores para las variables siguientes:

N = tamaño de la muestra a codificar.

J = nº de clases difusas equilibradas deseadas.

IORD = 1 ó 0, según que la muestra vaya a ser dada en orden creciente o no.

ISOL = 2 si deseamos que el programa calcule los intervalos solución (cuando existan) y construya una de las soluciones tomando automáticamente como p.r. de partida r_1 con el valor $(L_1+U_1)/2$.

= 1 calcula intervalos solución y completa una solución a partir de un p.r. que el usuario debe introducir: r_j.

= 0 el programa pide los intervalos solución y el p.r. de partida, completando una solución a partir de éste.

X(1)...X(J) valores de la muestra a codificar.

Cuando existe solución obtenemos en la salida valores para las siguientes variables entre otras:

L(j),U(j) cotas del Teorema 2.

IL(j),IU(j) nº de observaciones en la muestra, menosres que L(j) y U(j) resp. (nos da una idea de la posición de estos valores en la muestra).

R(j) p.r. solución.

IREF(j) nº de observaciones menores que el correspondiente p.r.

Ejemplo de una salida:

```
N               J            IORD          ISOL (Tarjeta de control):
40              8               1             2
```

(Lista de los 40 valores codificados, obtenidos en este caso simulando una Uniforme (0,1)):

```
0.014328063   0.046890438   0.050637305   0.063750446   0.07510668
0.1114971     0.1148793     0.1266391     0.1404701     0.1523901
0.1894552     0.1958491     0.2179943     0.2690665     0.2734967
0.2855063     0.3501683     0.3983895     0.4087101     0.4167263
0.4350125     0.4487864     0.4923264     0.5254094     0.5824481
0.6475641     0.6601760     0.6794111     0.6993963     0.7044111
0.7050844     0.7183182     0.7349077     0.7978860     0.7979252
0.8066970     0.8243976     0.8541732     0.9249521     0.9368497
```

INTERVALOS SOLUCION:

```
IL(j)    LL(j)          L(j)                    U(j)        UU(j)        IU(j)
 3   -0.01013754   0.057145 < R 1 < 0.111497   0.07510668      6
 5    0.11149710   0.075107 < R 2 < 0.125472   0.15239009      7
12    0.18945520   0.214193 < R 3 < 0.283241   0.27349669     15
15    0.28550631   0.273497 < R 4 < 0.376826   0.41672629     17
22    0.43501249   0.488573 < R 5 < 0.595200   0.58244812     25
25    0.64756411   0.622319 < R 6 < 0.705084   0.70441109     31
30    0.70508438   0.704411 < R 7 < 0.806697   0.79792517     36
35    0.80669701   0.797925 < R 8 < 0.905778   0.90577799     38
```

UNA SOLUCIÓN: (Gestionada mediante opción automática)

```
R 1 =   0.084321      IREF =  5
R 2 =   0.093302      IREF =  5
R 3 =   0.260888      IREF = 13
R 4 =   0.306293      IREF = 16
R 5 =   0.562589      IREF = 24
R 6 =   0.663799      IREF = 27
R 7 =   0.750043      IREF = 33
R 8 =   0.860331      IREF = 38
```

CONTINGENTES DIFUSOS (Para esta c.s.l.d.)

```
5.000000      5.000001      4.999999      4.999998
5.000002      5.000000      5.000000      5.000000
```

Detallamos en la siguiente table algunos tiempos de ejecución obtenidos al correr el programa para muestras simuladas de Uniformes (0,1) de diversos tamaños y diferente número de clases difusas.

N \ J	3	5	8	10	11	20	40	90
20	.97	.64	.78	.93	.95	1.27		
40	.95	.92	1.10	.84	1.00	1.00*	1.34	
100	1.09	1.16	1.14	1.12	1.05	.98*	.96*	.90*
1000	4.46	4.31	4.21	3.54	4.06	2.63*	2.51*	2.20*
10000		243.42			244.34			

Tiempos de ejecución en segundos de CPU
(* indica que no existe solución)

Hagamos notar finalmente que este algoritmo está diseñado para resolver el problema de encontrar estas c.s.l.d. equilibradas, con un criterio (a) de equilibrio basado en la distribución muestral $F_n(x)$:

$$T(o)=c.1_J, \text{ es decir } \int_R Z_j \, dF_n(x) = N/J \quad \forall j=1...J$$

pero puede ser también utilizado (método de Monte Carlo) para obtener soluciones tan fiables como deseemos para el problema de encontrar c.s.l.d. equilibradas, con un criterio (b) de equilibrio teórico:

$$\int_R Z_j \, dF(x) = N/J \quad \forall j=1...J,$$

no sujeto a variaciones en el muestreo como el (a). (aplicar para ello el algoritmo a muestras de tamaño N, tan grande como se desee, de una v.a. siguiendo una distribución $F(x)$).

APENDICE 1 * PASOS DEL ALGORITMO

Representamos abreviadamente con L_j===>U_k (análogamente U_j===> L_k, y sus negaciones con =≠=>) el siguiente hecho: $r_j > L_j$ OBLIGA a U_k. Representamos con # la orden " ir al comienzo del paso siguiente".

START

Paso 1: para j=1...J cargar L_j y U_j con los valores de partida:

$L_j(j)$ y $U_j(j)$; recordar que $L_1(1)=-\infty$ y $U_j(J)=\infty$;#.

Paso 2: U_2===> L_1 siempre. Cargo L_1; #.

Paso 3: a/ Si L_2===> U_3, cargo U_3; #.

b/ U_3===> L_2; cargo L_2;

b.0/ Si para este nuevo valor L_2 L_2=≠=> U_1; #.

b.1/ Si L_2===> U_1, cargo U_1; #.

.
.
.

Paso j: Si U_{j-1}===> L_j, Cargo L_j.

a/ Si L_{j-1}===> U_j, Cargo U_j; #.

b/ Si U_j===> L_{j-1}, Cargo L_{j-1}.

b.0/ Si para este nuevo valor L_{j-1}, L_{j-1}=≠=> U_{j-2}; #.

b.1/ Si L_{j-1}===> U_{j-2}, Cargo U_{j-2}.

b.1.0/ Si para este nuevo U_{j-2}, U_{j-2}=≠=> L_{j-3}; #.

b.1.1/ Si U_{j-2}===> L_{j-3}, Cargo L_{j-3}.

⋮

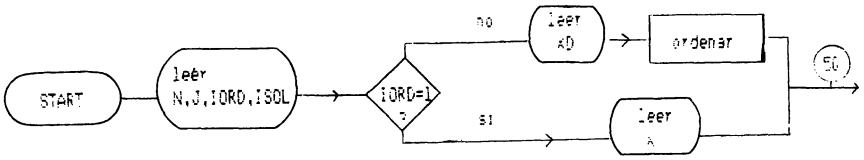
hasta llegar (salvo salida anterior) a L_1 ó U_1, según que j sea par o impar.

⋮

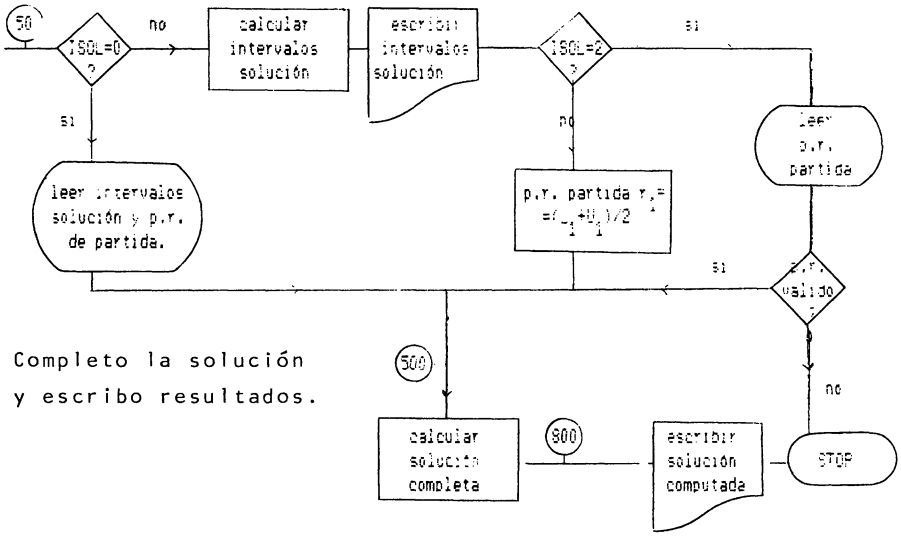
PASO J: #. STOP

APENDICE 2 # DIAGRAMA DE FLUJO ENTRE BLOQUES

Lectura de Tarjeta de control;
Carga y ordenación de observaciones.



Carga de intervalos solución
y de p.r. de partida para
completar la solución.



Completo la solución
y escribo resultados.

6. Bibliografia.

[1] CAZES P. (1980) [ANA. BLOCS II]. Les Cahiers de l'Analyse des Données. Vol V n°. 4 p. 387-403.

[2] GALLEGO J. (1980) Thèse 3-ème cycle, Paris.

[3] GALLEGO J. (1982) Codage Flou en Analyse des Correspondances. Les Cahiers de l'Analyse des Donées. Vol VIII n° 4 p. 413-440.

[4] GLZ. DE GARIBAY V. (1984). Análisis de la Varianza sobre una Codificación Difusa. Tesis Doctoral. Universidad Valladolid.

[5] GLZ. DE GARIBAY V. (1985) Funciones Paramétricas Estimables en Anova Difuso k-Criterio. XV Reunión de la SEIO. Asturias.

[6] GLZ. DE GARIBAY V. y BARBA L. (1985) Particiones Difusas Equiponderadas. Publ. Secc. de Matemáticas Univ. Valladolid n° 9. p 125-136.

[7] GUITONNEAU y ROUX (1977) [Le Genere Erodium] Les Cahiers de l'Analyse des Donées. Vol. II n° 1 p. 97-113.

[8] LEFOL (1979) Thèse 3-ème cycle. Paris.

[9] DUBOIS, D. y PRADE, H. (1980). Fuzzy Sets and Systems. New York. Academic Press.

Manuscript received in February 18, 1986, and in final form April 14, 1986.

Departamento de Estadística.
Facultad de Ciencias.
Universidad de Valladolid.
47071 VALLADOLID (SPAIN).