Lucrarea 8

METODE DE OPTIMIZARE

1. SCOPUL LUCRĂRII

Prezentarea unor algoritmi de optimizare, implementarea acestora într-un limbaj de nivel înalt (în particular, C) şi folosirea lor în rezolvarea unor probleme de electronică.

2. PREZENTAREA TEORETICĂ

Metodele de optimizare se clasifică, în raport cu problema care se pune pentru funcţia ţintă sau scop, astfel:

1. Metode ce determină expresia analitică a funcţiei, care aproximează cel mai bine o funcţie tabelată dată prin puncte;
2. Metode ce determină diferiţi parametri ai funcţiei scop (ţintă), pentru a obţine un extrem al funcţiei.

2.1. METODA CELOR MAI MICI PĂTRATE

Presupunem că avem o funcţie definită printr-un tabel de valori. Se mai spune că funcţia este dată implicit, deci nu i se cunoaşte forma în mod direct. Dacă tabelul este obţinut în urma unor măsurători fizice, atunci elementele lui pot avea erori. Elementele care diferă mult de celelalte pot fi eliminate. Problema care se pune este să determinăm funcţia analitică *care aproximează cel mai bine* datele din tabel sau curba care trece prin punctele tabelului.

2.2. COEFICIENTUL DE CORELAŢIE

Interpretarea erorilor de aproximaţie se poate face cu ajutorul coeficientului de corelaţie. Deşi se calculează un minim al erorii pătratice medii, există şi o interpretare grafică, din punct de vedere geometric. În final interesează cât de bună este potrivirea unei funcţii alese de noi peste porţiunea din graficul unei funcţii analitici necunoscute, pentru care cunoaştem doar o evoluţie locală, dată de setul de perechi (x_i, y_j) de la intrare.

2.2.1. CORELAȚIA EŞANTIOANELOR (evenimentelor eşantionate)

Dacă avem o serie de *n* măsurători ale variabilelor X şi Y, sub forma x_i si y_i, cu i = 1,2,…,n atunci pentru estimarea corelației dintre X şi Y se poate face pe baza *coeficientului de corelație produs-moment*, cunoscut sub numele de *coeficient Pearson*. Acesta mai este denumit şi coeficientul de corelație al eşantioanelor. Acesta este important mai ales dacă atât X cât şi Y sunt distribuite normal. Atunci coeficientul de corelație Pearson este cel mai bun estimat al corelației între X şi Y. Acest coeficient se scrie astfel:

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{(n-1)s_x s_y} \tag{8.1}$$

unde $\bar{x}$ şi $\bar{y}$ sunt *valorile medii* ale eşantioanelor x_i şi y_i, iar s_x şi s_y sunt *deviațiile standard* ale eşantioanelor x_i şi y_i. Sumarea se face după $i = \overline{1,n}$.

Desfăcând sumele relația anterioară se poate rescrie:

$$r_{xy} = \frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{\sqrt{n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2}\sqrt{n\sum_{i=1}^{n} y_i^2 - (\sum_{i=1}^{n} y_i)^2}} \tag{8.2}$$

Reamintim că valoarea absolută a coeficientului de corelație trebuie să fie mai mică sau egală cu 1. Cu toate că formula anterioară sugerează un algoritm într-un singur pas pentru calculul corelației eşantioanelor, este recunoscută instabilitatea numerică a unui asemenea algoritm.

2.2.2. INTERPRETAREA GEOMETRICĂ A CORELAȚIEI

Coeficientul de corelație poate fi privit şi drept cosinusul unghiului între doi vectori de eşantioane, generați de două valori aleatoare.

Atenție: această metodă funcționează doar cu date centrate, adică acele date care au fost deplasate cu media eşantioanelor, astfel încât au o medie aritmetică (*average*) egală cu 0. Există situații în practică în care se preferă un coeficient de corelație necentrat, dar care este incompatibil cu coeficientul Pearson.

Ca *exemplu*, să presupunem 5 țări pentru care produsul național brut este, respectiv: 1, 2, 3, 5 şi 8 milioane dolari. Presupunem că gradul de sărăcie al aceloraşi țări, păstrate în aceeaşi ordine este de `11%, 12%, 13%, 15%,` şi `18%`. Atunci, să

considerăm pe ***x*** şi ***y*** ca fiind doi vectori ordonaţi, cu 5 elemente, conţinând datele exprimate anterior:

x = (1, 2, 3, 5, 8)

şi

y = (0.11, 0.12, 0.13, 0.15, 0.18)

Urmând procedura de determinare a unghiului dintre doi vectori (pe baza produsului scalar), coeficientul de corelaţie necentrat va fi:

$$\cos\theta = \frac{\mathbf{x}\cdot\mathbf{y}}{\|\mathbf{x}\|\,\|\mathbf{y}\|} = \frac{2.93}{\sqrt{103}\sqrt{0.0983}} = 0.920814711 \tag{8.3}$$

Datele anterioare au fost alese astfel încât să existe o corelaţie foarte bună între ele:

y = 0.10 + 0.01 x

Coeficientul de corelaţie Pearson va trebui să fie exact 1. Centrând datele (deci deplasându-l pe ***x*** cu valoarea E(x) = 3.8 şi pe ***y*** cu valoarea E(y) = 0.138) obţinem:

x = (-2.8, -1.8, -0.8, 1.2, 4.2)

şi

y = (-0.028, -0.018, -0.008, 0.012, 0.042),

de unde:

$$\cos\theta = \frac{\mathbf{x}\cdot\mathbf{y}}{\|\mathbf{x}\|\,\|\mathbf{y}\|} = \frac{0.308}{\sqrt{30.8}\sqrt{0.00308}} = 1 \tag{8.4}$$

aşa cum ne aşteptam.

2.2.3. INTERPRETAREA MĂRIMII CORELAŢIEI

Mai mulţi autori oferă modalităţi de interpretare a coeficientului de corelaţie. De exemplu Cohen (1988) a sugerat următoarele interpretări ale corelaţiei, cu referire la studiile psihologice (Tabel 8.1):

Tabel 8.1

Corelaţia	Valori negative	Valori pozitive
Mic (*slabă*)	-0.29 to -0.10	0.10 to 0.29
Mediu (*normală*)	-0.49 to -0.30	0.30 to 0.49
Mare (*puternică*)	-1.00 to -0.50	0.50 to 1.00

După cum însuşi Cohen a observat, toate criteriile de acest fel sunt într-o oarecare măsură arbitrare şi nu ar trebui luate în sens strict. Aceasta deoarece interpretarea unui coeficient de corelaţie depinde de context şi de scopurile avute în vedere. O corelaţie de

0.9 poate fi foarte scăzută dacă cineva verifică legile fizicii folosind instrumente de înaltă calitate, dar poate fi foarte ridicată în contextul ştiinţelor sociale unde apar influenţe datorate unor factori complicaţi.

Pentru *problemele de regresie* la care ne referim în laborator, interpretarea se poate face conform tabelului indicat de Cohen, pentru că nu există factori esenţiali care influenţează datele preluate, alţii decât erorile obişnuite de măsurare şi interpretare, pentru care se respectă regulile specifice de eliminare a experimentelor eronate, de rotunjire a valorilor conform cu erorile de calcul acceptate sau de reprezentare normalizată (pentru valorile foarte mici sau foarte mari).

3. METODE DE REGRESIE

3.1.1. REGRESIA LINIARĂ

Se consideră funcţia tabelată:

Tabelul 8.2

x	x_1	x_2	x_3	x_4	…..	x_m
y	y_1	y_2	y_3	y_4	…..	y_m

unde m reprezintă *numărul de măsurători* sau de valori ale funcţiei. Se cere să se determine funcţia liniară de forma generală

$$y = ax + b \tag{8.1}$$

care să aproximeze cel mai bine funcţia tabelată ***astfel ca eroarea să fie minimă***.

$$a = \frac{m\sum_{i=1}^{m} x_i y_i - \sum_{i=1}^{m} x_i \sum_{i=1}^{m} y_i}{m\sum_{i=1}^{m} x_i^2 - \left(\sum_{i=1}^{m} x_i\right)^2} \tag{8.2}$$

$$b = \frac{\sum_{i=1}^{m} x_i^2 \sum_{i=1}^{m} y_i - \sum_{i=1}^{m} x_i \sum_{i=1}^{m} x_i y_i}{m\sum_{i=1}^{m} x_i^2 - \left(\sum_{i=1}^{m} x_i\right)^2} \tag{8.3}$$

3.1.1.1. Algoritm 8.1. Regresia liniară

```
void Reg_Lin
(
   întreg m,    // numărul experienţelor
   real x[ ],   // vectorul absciselor funcţiei
   real y[ ],   // vectorul ordonatelor funcţiei
   real *pA,    // adresa coeficientului lui x (pointer)
   real *pB          // adresa termenului liber (pointer)
)
```

```
{ // declararea şi definirea variabilelor locale
   real sx ;   // variabila pentru suma absciselor
   real sy ;   // variabila pentru suma ordonatelor
   real sxx ;  // variabila pentru sumele de forma x^2
   real sxy,   // variabila pentru sumele de forma xy
   întreg i ;  // contor pentru ciclurile 'for()'.

   // corpul de instrucţiuni al funcţiei
   sx= 0 ; sy = 0 ; sxy = 0 ; sxx = 0;   // iniţializare cu 0
   pentru i= 1÷ m
   {
          sx = sx + x[i] ;
          sy = sy + y[i] ;
          sxy = sxy + x[i]*y[i] ;
          sxx = sxx + x[i]*x[i] ;
   }
   *pA = (m·sxy-sx·sy) / (m·sxx-sx·sx) ;    *pB = (·sxx·sy-sx·sxy) / (m·sxx-sx·sx) ;
}
```

Obs.:

1) Am marcat faptul că funcţia nu întoarce nici o valoare prin cuvântul-cheie '*void*', la fel ca în C.
2) Dimensiunea vectorilor folosiţi de metodă se stabileşte funcţie de numărul 'm' al punctelor experimentale considerate.
3) Denumirile ultimelor două argumente ale funcţiei respectă o convenţie de notaţie – şi anume *notaţia ungară* – în care se utilizează şi anumite indicii literale despre tipul variabilei. Aceste indicii literale *preced* numele variabilei. Aceste două argumente fiind de tip referinţă (deci pointer) denumirile variabilelor încep cu litera mică *p*, ca sugestie pentru pointeri.

3.1.2. REGRESIA POLINOMIALĂ

Dacă regresia liniară nu este satisfăcătoare se caută o funcţie polinom de un anumit grad (2, 3, 4…*n*).

Fie funcţia dată în tabelul (8.1). Considerăm că reprezentarea grafică a acestei funcţii se aseamănă foarte mult cu curba unui polinom de gradul *n* de forma:

$$y = a_n x^n + a_{n-1} x^{n-1} + ... + a_1 x + a_0 \tag{8.4}$$

Coeficienţii polinomului se determină cu ajutorul sistemului:

$$\begin{cases} ma_0 + \left(\sum_{i=1}^{m} x_i\right)a_1 + \left(\sum_{i=1}^{m} x_i^2\right)a_2 + \cdots + \left(\sum_{i=1}^{m} x_i^k\right)a_k + + \left(\sum_{i=1}^{m} x_i^n\right)a_n = \sum y_i \\ \left(\sum_{i=1}^{m} x_i\right)a_0 + \left(\sum_{i=1}^{m} x_i^2\right)a_1 + \left(\sum_{i=1}^{m} x_i^3\right)a_2 + \cdots + \left(\sum_{i=1}^{m} x_i^{k+1}\right)a_k + \cdots + \left(\sum_{i=1}^{m} x_i^{n+1}\right)a_n = \sum x_i y_i \\ \hline \left(\sum_{i=1}^{m} x_i^p\right)a_0 + \left(\sum_{i=1}^{m} x_i^{p+1}\right)a_1 + \left(\sum_{i=1}^{m} x_i^{p+2}\right)a_2 + \cdots + \left(\sum_{i=1}^{m} x_i^{p+k}\right)a_k + \cdots + \left(\sum_{i=1}^{m} x_i^{p+n}\right)a_n = \sum x_i^p y_i \\ \left(\sum_{i=1}^{m} x_i^n\right)a_0 + \left(\sum_{i=1}^{m} x_i^{n+1}\right)a_1 + \left(\sum_{i=1}^{m} x_i^{n+2}\right)a_2 + \cdots + \left(\sum_{i=1}^{m} x_i^{n+k}\right)a_k + \cdots + \left(\sum_{i=1}^{m} x_i^{2n}\right)a_n = \sum x_i^n y_i \end{cases} \tag{8.5}$$

Sistemul obținut este un sistem liniar în necunoscutele a_0, a_1, a_2,..., a_n şi se rezolvă cu una dintre metodele cunoscute în ***Lucrarea 3 - Metode pentru rezolvarea sistemelor liniare de ecuații***.

3.1.2.1. Algoritmul 8.2.Regresia polinomială

```
întreg Reg_Pol
(
   întreg m,   // numărul de puncte cunoscute ale funcţiei
   real X[ ],  // vectorul absciselor funcţiei
   real Y[ ],  // vectorul ordonatelor funcţiei
   întreg n,   // gradul polinomului de regresie
   real *coef     // adresa de început a vectorului
                  //coeficienţilor polinomului de regresie.
)
{ // declararea şi definirea variabilelor locale
   întreg i, j, k ;  // contoare
   real A[N][N] ;       // matrice utilizată pentru
                        //rezolvarea sistemului (8.5).
   real B[N] ; //vectorul termenilor liberi (sistemul (8.5))

   // corpul de instrucţiuni al funcţiei
   pentru i= 1÷ n+1
   {
   pentru j= i ÷ n+1
   {
   A[i][j] = 0 ;
   pentru k= 1÷ n   A[i][j] = A[i][j] + pow( x[k], i+j-2) ;
   A[j][i] = A[i][j] ;
   }
   B[i] = 0 ;
   pentru  k= 1÷ n    B[i] = B[i] + y[k] * pow( x[k], i-1) ;
   }
   // rezolvă sistemul 7.5
   dacă ( Gauss(n+1,A,B,coef) !=0 ) returnează True ;
   altfel returnează False ;
}
```

Obs.:

1) Am marcat faptul că funcția nu întoarce nici o valoare prin cuvântul-cheie '*void*', la fel ca în C;
2) Dimensiunea 'N' a vectorilor folosiți de metodă se stabilește funcție de numărul '*m*' al punctelor considerate, având în vedere că indexul minim al vectorilor în C este *0*;
3) Funcția '*Gauss( )*' la care se face referire este cea din ***Lucrarea 3 - Metode pentru rezolvarea sistemelor liniare de ecuații***.

3.1.3. REGRESIA HIPERBOLICĂ

Fie funcția numerică dată în tabelul 8.1. Se pune problema determinării funcției hiperbolice: $$y = \frac{1}{ax+b} \tag{8.6}$$ care aproximează cel mai bine funcția numerică. Valorile lui *a* şi *b* sunt:

$$a = \frac{(\sum_{i=1}^{m} x_i)(\sum_{i=1}^{m} \frac{1}{y_i}) - m\sum_{i=1}^{m} \frac{x_i}{y_i}}{\left(\sum_{i=1}^{m} x_i\right)^2 - m\sum_{i=1}^{m} x_i^2} \tag{8.7}$$

$$b = \frac{(\sum_{i=1}^{m} \frac{1}{y_i})(\sum_{i=1}^{m} x_i^2) - (\sum_{i=1}^{m} \frac{x_i}{y_i})(\sum_{i=1}^{m} x_i)}{m\sum_{i=1}^{m} x_i^2 - \left(\sum_{i=1}^{m} x_i\right)^2} \tag{8.8}$$

Înlocuim valorile lui *a* şi *b* în funcția hiperbolică (8.6) şi se obține funcția care aproximează cel mai bine funcția numerică din tabelul 8.1.

Pentru a liniariza rezolvarea sistemului s-a lucrat cu inversele valorilor lui *y*, adică *1/y*, motiv pentru care sumele *sy* şi *sxy* conțin valorile inversate pentru *y*.

3.1.3.1. Algoritm 8.3. Regresia hiperbolică

```
void Reg_Hip
(
întreg m,       // numărul experiențelor
real x[ ],      // vectorul absciselor funcției numerice
real y[ ],      // vectorul ordonatelor funcției numerice
real *pA        // adresa coeficientului lui a al funcției
                //hiperbolice
real *pB        // adresa coeficientului lui b al funcției
                //hiperbolice
)
```

```
{ // declararea  şi definirea variabilelor locale
real sx;        //  variabila ce conţine sumele absciselor
real sy ;       //  variabila ce conţine suma inverselor
                // ordonatelor.
real sxx ;      // variabila ce conţine pătratul absciselor
real sxy ;      // variabila ce conţine suma raportului
                // dintre abscise şi ordonate..
întreg i ;      // contor

   // corpul de instrucţiuni al funcţiei
   sx = 0 ; sy = 0 ; sxy = 0 ; sxx = 0 ;
   pentru i= 1÷m
   {      // urmează un corp de instrucţiuni, deci se
          // foloseşte acolada deschisă.
          sx = sx + x[i] ;
          sy = sy+ 1/y[i] ;
          sxy= sxy+ x[i]/y[i] ;
          sxx = sxx + x[i]*x[i] ;
   }
   *pA = (m·sxy − sx·sy)/(m·sxx − sx·sx) ; *pB = (·sxx·sy-sxy·sx)/(m·sxx-sx·sx) ;
}
```

Obs.:

1) Am marcat faptul că funcţia nu întoarce nici o valoare prin cuvântul-cheie '*void*', la fel ca în C;
2) Dimensiunea vectorilor folosiţi de metodă se stabileşte funcţie de numărul '*m*' al punctelor considerate, având în vedere că indexul minim al vectorilor în C este *0*;
3) Denumirile ultimelor două argumente ale funcţiei respectă o convenţie de notaţie – şi anume *notaţia ungară* – în care se utilizează şi anumite indicii literale despre tipul variabilei. Aceste indicii literale *preced* numele variabilei. Aceste două argumente fiind de tip referinţă (deci pointer) denumirile variabilelor încep cu litera mică *p*, ca sugestie pentru pointeri.

3.1.4. REGRESIA EXPONENŢIALĂ

Dacă se constată că funcţia numerică din tabelul 8.1, reprezentată grafic, se aseamănă foarte mult cu o exponenţială, atunci se consideră funcţia exponenţială, de forma generală:

$$y=a\cdot b^x \tag{8.9}$$

în care a şi b sunt constante pozitive.

Pentru un calcul comod al constantelor a şi b se logaritmează funcția (8.9):

$$\ln y = \ln a + x \ln b \tag{8.10}$$

şi se determină constantele astfel încât eroarea pătratică de ansamblu să fie *minimă*.

$$a = \exp\left\{\frac{\left[(\sum_{i=1}^{m} \ln y_i)(\sum x_i^2) - (\sum_{i=1}^{m} x_i)(\sum_{i=1}^{m} \ln y_i x_i)\right]}{\left[m\sum_{i=1}^{m} x_i^2 - \left(\sum_{i=1}^{m} x_i\right)^2\right]}\right\} \tag{8.11}$$

$$b = \exp\left\{\frac{\left[m\sum_{i=1}^{m} x_i \ln y_i - (\sum_{i=1}^{m} x_i)(\sum_{i=1}^{m} \ln y_i)\right]}{\left[m\sum_{i=1}^{m} x_i^2 - \left(\sum_{i=1}^{m} x_i\right)^2\right]}\right\} \tag{8.12}$$

3.1.4.1. Algoritm 8.4. Regresia exponențială

```
void Reg_Exp
(
   întreg m,   // numărul absciselor funcţiei
   real x[ ],  // vectorul absciselor funcţiei numerice
   real y[ ],  // vectorul ordonatelor funcţiei numerice
   real *pA,   // adresa coeficientului lui a al funcţiei
               // exponenţiale.
   real *pB    // adresa coeficientului lui b al funcţiei
               // exponenţiale.
)
{ // declararea şi definirea variabilelor locale
real sx  ;     // suma  absciselor
real sxx ;     // suma pătratelor absciselor
real sy ;      // suma logaritmilor din ordonate
real sxy ;     // suma absciselor înmulţite cu logaritmul
               // ordonatelor.
întreg i ;     // contor pentru bucla 'for()'.
//  corpul de instrucţiuni al funcţiei
   sx= 0 ; sy = 0 ; sxy = 0 ; sxx = 0 ;  // iniţializare
   pentru i= 1÷ m
   {
         sx = sx + x[i] ;
         sxx = sxx + x[i]*x[i] ;
         sy = sy+log( y[i]) ;
         sxy = sxy+ x[i] * log( y[i]) ;
   }
   *pA = exp( (sy · sxx - sx · sxy) / (m · sxx - sx · sx) ) ;
   *pB = exp( (m sxy - sx sy) / (m · sxx - sx · sx) ) ;
}
```

Obs.:

1) Am marcat faptul că funcţia nu întoarce nici o valoare prin cuvântul-cheie '*void*', la fel ca în C;
2) Dimensiunea vectorilor folosiţi de metodă se stabileşte funcţie de numărul '*m*' al punctelor experimentale considerate, având în vedere că indexul minim al vectorilor în C este *0*;
3) Denumirile ultimelor două argumente ale funcţiei respectă o convenţie de notaţie – şi anume *notaţia ungară* – în care se utilizează şi anumite indicii literale despre tipul variabilei. Aceste indicii literale *preced* numele variabilei. Aceste două argumente fiind de tip referinţă (deci pointer) denumirile variabilelor încep cu litera mică *p*, ca sugestie pentru pointeri.

3.1.5. REGRESIA GEOMETRICĂ

Dacă funcţia numerică din tabelul 8.1 se aseamănă cu o funcţie de tip geometric, vom căuta să determinăm funcţia de forma:

$$y = ax^b \tag{8.13}$$

care să aproximeze cel mai bine funcţia numerică. Adică aceeaşi condiţie de eroare pătratică, de ansamblu, minimă.

Şi în această situaţie se recurge la o logaritmare a relaţiei, pentru a aduce rezolvarea în domeniul liniar. Avem:

$$lny = ln(a) + b*ln(x) \tag{8.13'}$$

Se obţin următoarele valori pentru a şi b:

$$a = \exp\frac{\left(\sum_{i=1}^{m}\ln(y_i)\right)\left(\sum_{i=1}^{m}\ln^2(x_i)\right)-\left(\sum_{i=1}^{m}\ln\ (x_i)\right)\left(\sum_{i=1}^{m}\ln\ (x_i)\ln(y_i)\right)}{m\left(\sum_{i=1}^{m}\ln^2(x_i)\right)-\left(\sum_{i=1}^{m}\ln\ (x_i)\right)^2} \tag{8.14}$$

$$b = \frac{m\left(\sum_{i=1}^{m}\ln(y_i)\ln(x_i)\right)-\left(\sum_{i=1}^{m}\ln\ (x_i)\right)\left(\sum_{i=1}^{m}\ln(y_i)\right)}{m\left(\sum_{i=1}^{m}\ln^2(x_i)\right)-\left(\sum_{i=1}^{m}\ln\ (x_i)\right)^2} \tag{8.15}$$

Cu aceste valori funcţia $y = ax^b$ aproximează cel mai bine funcţia tabelată (8.1) care se aseamănă cel mai bine cu o funcţie de tip geometric.

3.1.5.1. Algoritmul 8.5. Regresia geometrică

```
void Reg_Geo
(
 întreg m,     // numărul de abscise
 real x[ ],    // vectorul absciselor funcţiei numerice
 real y[ ],    // vectorul ordonatelor funcţiei numerice
 real *pA,      // adresa coeficientului lui a al funcţiei
                //geometrice
 real *pB       // adresa coeficientului lui b al funcţiei
                //geometrice
)
{ // declararea şi definirea variabilelor locale
real sx ;       // suma logaritmilor absciselor
real sy ;       // suma logaritmilor ordonatelor
real sxy ;      // suma produsului intre logaritmii
                //absciselor şi logaritmilor ordonatelor
real sxx  ;     // suma pătratelor logaritmilor absciselor
întreg i ;      // contor

   // corpul de instrucţiuni al funcţiei
   sx = 0 ; sy = 0 ; sxx = 0 ; sxy= 0 ;  // iniţializare
   pentru i= 1÷ m
   {
         sx = sx + log(x[i]) ;
         sxx = sxx + log(x[i])*log(x[i]) ;
         sy = sy + log (y[i]) ;
         sxy = sxy + log(x[i])* log(y[i]) ;
   }
```

$$*pA = \exp\left(\frac{s_y \cdot s_{xx} - s_x \cdot s_{xy}}{m \cdot s_{xx} - s_x \cdot s_x}\right); \qquad *pB = \frac{m \cdot s_{xy} - s_x \cdot s_y}{m \cdot s_{xx} - s_x \cdot s_x};$$

```
}
```

Obs.:

1) Am marcat faptul că funcţia nu întoarce nici o valoare prin cuvântul-cheie '*void*', la fel ca în C;
2) Dimensiunea vectorilor folosiţi de metodă se stabileşte funcţie de numărul '*m*' al punctelor experimentale considerate, având în vedere că indexul minim al vectorilor în C este *0*;
3) Denumirile ultimelor două argumente ale funcţiei respectă o convenţie de notaţie – şi anume *notaţia ungară* – în care se utilizează şi anumite indicii literale despre tipul variabilei. Aceste indicii literale *preced* numele variabilei. Aceste două argumente fiind de tip referinţă (deci pointer) denumirile variabilelor încep cu litera mică *p*, ca sugestie pentru pointeri.

3.1.6. REGRESIA TRIGONOMETRICĂ

Se consideră funcţia numerică dată în tabelul 8.1 şi funcţia trigonometrică de forma:

$$y = a + b \cdot \cos(\omega \cdot x) \tag{8.16}$$

care aproximează cel mai bine funcţia numerică.

Valorile lui *a* şi *b* sunt:

$$a = \frac{\left(\sum_{i=1}^{m} y_i\right)\left(\sum_{i=1}^{m} \cos^2 \omega x_i\right) - \left(\sum_{i=1}^{m} \cos \omega x_i\right)\left(\sum_{i=1}^{m} y_i \cos \omega x_i\right)}{m\left(\sum_{i=1}^{m} \cos^2 \omega x_i\right) - \left(\sum_{i=1}^{m} \cos \omega x_i\right)^2} \tag{8.17}$$

$$b = \frac{m\left(\sum_{i=1}^{m} y_i \cos \omega x_i\right) - \left(\sum_{i=1}^{m} \cos \omega x_i\right)\left(\sum_{i=1}^{m} y_i\right)}{m\left(\sum_{i=1}^{m} \cos^2 \omega x_i\right) - \left(\sum_{i=1}^{m} \cos \omega x_i\right)^2} \tag{8.18}$$

$\omega = 2 \cdot \pi \cdot f = \frac{2 \cdot \pi}{T}$ unde f - frecvenţa, ω - pulsaţia, iar T - perioada.

ω se stabileşte funcţie de periodicitatea funcţiei numerice date.

3.1.6.1. Algoritmul 8.6. Regresia trigonometrică

```
void Reg_Tri
(
   întreg m,    // numărul  absciselor funcţiei numerice
   real x[ ],   // vectorul absciselor funcţiei numerice
   real y[ ],   // vectorul ordonatelor funcţiei numerice
   real *pA, // adresa coeficientului lui 'a' din expresia
             //funcţiei trigonometrice
   real *pB  // adresa coeficientului lui 'b' din expresia
             //funcţiei trigonometrice
)
{ // declararea şi definirea variabilelor locale
real ω ;        // pulsaţia ( =2*PI*T)
real s y ;      // suma ordonatelor funcţiei numerice
real scos 2x: ; // suma pătratelor cosinusurilor din abscise
real scos x  ;  // suma cosinusurilor din abscise
real sycosx ;   // suma produsului ordonatelor cu cosinusul
                // absciselor  corespunzătoare.
întreg i ;      // variabila de control a ciclului 'for()' şi
                //index de vector.

// corpul de instrucţiuni al funcţiei
         // iniţializarea sumelor
   s y = 0 ; scos 2x = 0 ; scos x = 0 ; sycos x = 0 ;
   pentru i= 1÷ m
```

```
    {
    s y = s y + y[i] ;
    scos 2x = scos 2x + cos(ω xi) * cos(ω xi) ;
    scos x = scos x + cos(ω xi) ;
    sycos x = sycos x + yi* cos(ω xi) ;
    }
```

$$*pA = \frac{s_y \cdot s_{\cos 2x} - s_{\cos x} \cdot s_{y\cos x}}{m \cdot s_{\cos 2x} - s_{\cos x} \cdot s_{\cos x}} \; ; \qquad *pB = \frac{m \cdot s_{y\cos x} - s_{\cos x} \cdot s_y}{m \cdot s_{\cos 2x} - s_{\cos x} \cdot s_{\cos x}} \; ;$$

```
}
```

Obs.:

1) Am marcat faptul că funcția nu întoarce nici o valoare prin cuvântul-cheie '*void*', la fel ca în C;
2) Dimensiunea vectorilor folosiți de metodă se stabilește funcție de numărul '*m*' al punctelor experimentale considerate, având în vedere că indexul minim al vectorilor în C este *0*;
3) Denumirile ultimelor două argumente ale funcției respectă o convenție de notație – şi anume *notația ungară* – în care se utilizează şi anumite indicii literale despre tipul variabilei. Aceste indicii literale *preced* numele variabilei. Aceste două argumente fiind de tip referință (deci pointer) denumirile variabilelor încep cu litera mică *p*, ca sugestie pentru pointeri.

3.1.7. REGRESIA MULTIPLĂ

Această regresie se referă la funcțiile de mai multe variabile.

Considerând o funcție numerică de *n* variabile $y = f(x_1, x_2, \ldots, x_n)$ şi un tip de funcție analitică de *n* variabile care se aseamănă cu cea numerică, se pune problema determinării funcției analitice astfel ca ea *să aproximeze cel mai bine* funcția numerică.

Vom considera funcția numerică dată în figura 8.1.

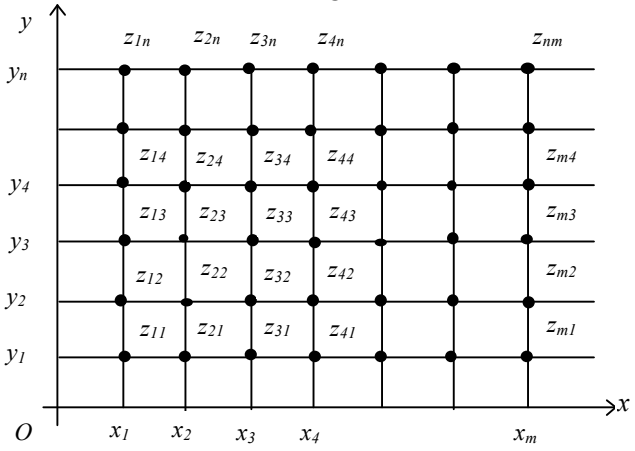

Fig. 8.1 - Reprezentarea funcției multiple.

Considerăm că punctele $z_1, z_2, ..., z_n$ se află aproximativ pe un plan:

$$z = Ax + By + C \tag{8.19}$$

Se pune problema determinării acestui plan, adică a valorilor constantelor *A, B, C* astfel ca planul să aproximeze cel mai bine o funcția numerică de două variabile dată.

Valorile lui *A, B, C* sunt soluțiile sistemului:

$$\begin{cases} mnC+\left(\sum_{i=1}^{m} x_i\right)nA+\left(\sum_{j=1}^{n} y_j\right)mB=\sum_{i=1}^{m}\sum_{j=1}^{n} z_{ij} \\ \left(\sum_{i=1}^{m} x_i\right)nC+\left(\sum_{i=1}^{m} x_i^2\right)nA+\sum_{i=1}^{m}\sum_{j=1}^{n} x_i y_j=\sum_{i=1}^{m}\sum_{j=1}^{n} x_i z_{ij} \\ \left(\sum_{j=1}^{n} y_j\right)mC+\left(\sum_{i=1}^{m}\sum_{j=1}^{n} x_i y_j\right)A+\left(\sum_{j=1}^{n} y_j^2\right)mB=\sum_{i=1}^{m}\sum_{j=1}^{n} y_j z_{ij} \end{cases} \tag{8.20}$$

Utilizând una dintre metodele numerice de rezolvare a sistemelor liniare din capitolul 3 se determină *A, B, C,* deci planul căutat.

3.1.7.1. Algoritmul 8.7. Regresia multiplă

```
întreg Reg_Mul
(
   întreg m,   // numărul de argumente pe Ox
   întreg n,   // numărul de argumente pe Oy
   real x[ ],  // vectorul argumentelor pe axa Ox a funcţiei
               // numerice.
   real y[ ],  // vectorul argumentelor pe axa Oy a funcţiei
               // numerice.
   real z[ ],  // vectorul valorilor funcţiei numerice
   real *coef     // adresa de început a vectorului
                  //coeficienţilor planului.
)
{ // declararea şi definirea variabilelor locale
întreg i, j ;  // indici
real A[4][N] ; // matricea sistemului (8.20);
real B[4] ;    // vectorul termenilor liberi ai sistemului
               //(8.20)
// corpul de instrucţiuni al funcţiei
   A[1][1] = m*n ;   A[1][2] = 0 ;
   pentru  i= 1÷ m   A[1][2] = A[1][2] + n*x[i] ;
   A[2][1] = A[1][2] ;     A[1][3] = 0 ;
   pentru  i= 1÷ n  A[1][3] = A[1][3] + m*y[i] ;
   A[3][1] = A[1][3] ;     A[2][3]=0 ;
   B[1] = 0 ;  B[2] = 0 ;  B[3 ] = 0 ;
   pentru i= 1÷ m
         pentru j= 1÷ n
         {
         A[2][3] = A[2][3] + x[i]*y[j] ;
         B[1] = B[1] + z[i][j] ;
```

```
            B[2] = B[2] + x[i]*z[i][j] ;
            B[3] = B[3] + y[j]*z[i][j] ;
            }
      A[3][2]=A[2][3];  A[2][2]=0 ;
      pentru i= 1÷ m  A[2][2] = A[2][2] + n*x[i]*x[i] ;
      A[3][3] = 0 ;

// rezolvă sistemul (8.20)
         pentru i= 1÷ n  A[3][3] = A[3][3] + m*y[i]*y[i] ;
         dacă ( Gauss( 3, A, B, coef) != 0 ) returnează True ;
         altfel returnează False ;
}
```

Obs.:

1) Funcția '*Gauss( )*' la care se face referire este cea din ***Lucrarea 3 - Metode pentru rezolvarea sistemelor liniare de ecuații***.
2) Dimensiunile matricelor şi vectorilor folosiți de metodă se stabileşte funcție de numerele '*m*' şi '*n*', ale punctelor experimentale considerate pe fiecare axă de coordonate.

3.2. OPTIMIZAREA NELINIARĂ FĂRĂ RESTRICȚII

3.2.1. METODE ALEATOARE DE CĂUTARE

Procesul este iterativ şi pentru fiecare etapă se generează o succesiune de numere aleatoare având o densitate de probabilitate uniformă în domeniul de variație admis, cu care se modifică componentele vectorului.

Metoda drumului aleator, una dintre metodele des utilizate, constă în modificarea vectorului de poziție pentru noul punct astfel:

$$X_{k+1} = X_k + \alpha.r \tag{8.21}$$

unde *r* este vectorul unitate care are direcția aleatoare, X_k este vectorul de poziție al punctului anterior şi α un scalar. Se calculează valoarea funcției în noul punct şi se compară cu valoarea funcției în vechiul punct. Dacă, după un anumit număr de iterații valoarea lui *F* nu se micşorează, se reduce scalarul α până ce valoarea lui devine mai mică decât o eroare de calcul dată, $er > 0$.

3.2.1.1. Algoritmul 8.8. Metoda drumului aleatoriu

```
real Min_Aleator
(
   Adr_functie,      // Adresa unei funcții reale de două
                  // variabile reale(pointer la o funcție)
   real x0[ ],       // vectorul de start
   real eps          // eroarea de calcul
)
```

```
{ // declararea şi definirea variabilelor locale
real gama0 ;          // scalar
întreg contor ;       // indice
real minim ;          // minimul dintre valorile funcţiilor
real val_curentă ;    // valoarea minimă curentă a funcţiei
real xcurent[ ] ;     // valoarea anterioară şi curentă a
                      // abscisei punctului.
întreg  maxiter ;     // numărul maxim de iteraţii

// corpul de instrucţiuni al funcţiei
   randomize( );  // macro ce pregăteşte folosirea funcţiei
                  // 'random()' prin iniţializarea
                  //generatorului de  numere aleatoare.
   gama0 = 1 ;
   cât timp ( gama0 >eps)
   {      // cât timp
   contor = 0 ;
   minim = *Adr_functie( x0[0], x0[1]) ;
   execută
   {
   x_curent[0]=x0[0] + gama0*pow(-1, random(101) mod 2);
   x_curent[1]=x0[1] + gama0*pow(-1, random(101) mod 2);
   val_curentă = *Adr_functie( x_curent[0], x_curent[1]) ;
   incrementează contor ;
   }
   cât timp( (contor <= maxiter) SI (val_curentă >= minim));
   dacă ( val_curenta<minim)
   {            // dacă
   minim = val_curentă ;
   x0[0] = x_curent[0] ;
   x0[1] = x_curent[1] ;
   }
   dacă ( contor > maxiter)  gama0 = gama0 - 0.001 ;
   } // cât timp
   returnează  minim ;
}
```

Obs.:

Notaţiile '*Adr_functie*' şi '***Adr_functie*' sunt simbolice. Ele reprezintă, respectiv, adresa unei funcţii (deci un pointer la o funcţie) şi apelul unei funcţii, indirect, prin pointerul al aceasta. Pentru lămuriri consultaţi *Anexa B* - "*Noţiuni de C necesare desfăşurării lucrărilor de laborator*".

3.2.2. METODA CĂUTĂRII UNIDIMENSIONALE

Această metodă determină minimul unei funcţii $F(X)$ prin modificarea componentelor vectorului X pe rând, începând cu prima componentă. Se modifică prima componentă atâta timp cât se obţine o valoare mai mică a funcţiei în punctul curent decât în cel precedent. Când această condiţie nu mai este îndeplinită se trece la

următoarea componentă a vectorului X şi se procedează la fel până se baleiază toate componentele vectorului considerând, în acest caz, că s-a realizat un ciclu. Se poate începe un nou ciclu luând ca punct inițial minimul obținut. La fiecare iterație se determină un nou punct funcție de precedentul astfel:

$$X_{k+1} = X_k + pas \cdot E_k \tag{8.22}$$

unde E_k reprezintă vectorul E_k *=(0,0, ...,1,...,0)*, cifra 1 fiind pe locul *k, k=0,1,...,n*, iar *pas* reprezintă pasul cu care se modifică componentele. După un ciclu încheiat se trece la un nou ciclu cu vectorul inițial cel pentru care s-a obținut minimul în ciclul încheiat (precedent), luând un nou pas mai mic, obținut din pasul precedent înmulțit cu o rație subunitară. Se continuă iterația până când *pas* < ε sau numărul de iterații depăşesc un număr maxim dat. ε reprezintă eroarea de calcul a punctului de minim.

3.2.2.1. Algoritmul 8.9. Metoda căutării unidimensionale

```
real Min_Unidimensional
(
   Adr_functie,   // Adresa unei funcţii reale de două
                  //variabile reale( pointer la o funcţie)
   real x0[ ], // vectorul de start
   real eps    // eroarea de calcul
)
{ // declararea şi definirea variabilelor locale
real pas ;     // pasul de modificare al coordonatelor
întreg i ;     // contor
real min_curent,min_prec;  // minimul curent respectiv prec.
intreg nrp ;   // numărul de paşi
real stânga = -1 ;
real dreapta = 1 ;
real direcţie ;// direcţia de deplasare: stânga sau dreapta.

   // corpul de instrucţiuni al funcţiei
   pas = 1 ;
   min_curent = *Adr_functie(x0[0], x0[1]) ;
   execută
   {
   pentru i= 0..1
   { // pentru( )
   nrp = 0 ;
   direcţie = dreapta ;
   execută
   {
   min_prec = min_curent ;
   incrementează nrp ;
   x0[i]=x0[i]+directie*pas;
   min_curent = *Adr_functie( x0[0], x0[1]) ;
   dacă ( (nrp=1) ŞI (min_curent > min_prec)
   {
         x0[i] = x0[i]-2*pas ;
      min_curent = *Adr_functie(x0[0], x0[1]) ;
```

```
            incrementează nrp ;
            direcţie = stânga ;
      }
      } cât timp( min_curent <= min_prec) ;
      x0[i] = x0[i] - direcţie*pas ;
      min_curent = *Adr_functie( x0[0], x0[1]) ;
    } // pentru( )
    pas = pas - 0.0001 ;
  } cât timp( pas >= eps) ;
    returnează  min_curent ;
}
```

Obs.:

Notaţiile '*Adrf*' şi '*(*Adrf)*' sunt simbolice. Ele reprezintă, respectiv, adresa unei funcţii (deci un pointer la o funcţie) şi apelul indirect al unei funcţii, prin intermediul pointerului la aceasta. Pentru lămuriri consultaţi *Anexa B* - "*Noţiuni de C necesare desfăşurării lucrărilor de laborator*".

4. DESFĂŞURAREA LUCRĂRII

4.1. ÎNTREBĂRI

1. Care este diferenţa dintre metodele de optimizare şi cele de interpolare?
2. O problema de optimizare se va soluţiona întotdeauna cu succes deplin pentru orice valori ale unei funcţii?
3. Prezentaţi pe scurt principiul de lucru al unei metode de regresie.
4. Ce poate influenţa precizia de calcul a oricărei metode de optimizare?
5. Cum se defineşte eroarea pătratică medie? Este aceasta o funcţie de una sau de mai multe variabile?
6. Numiţi câteva utilizări practice ale operaţiei de regresie.

4.2. PROBLEME LABORATOR

1. Să se implementeze metodele 2.1.1—2.1.7 şi 2.2.1—2.2.2 în limbajul ANSI C.

2. Se dau datele experimentale din tabelul 8.2:

Tabelul 8.2.

X	1	2	3	4	5	6	7	8	9
Y	1.44	1.728	2.0736	2.4883	2.9859	3.5831	4.2998	5.1597	6.191

Să se determine funcţia care aproximează cel mai bine funcţia tabelată cu metodele 2.1.1.— 2.1.5.

3. Se dă funcţia: $z = x^2 + y^2 - 2x - 4y + 3$. Să se determine minimul funcţiei cu metodele *drumului aleator* şi a *căutării unidimensionale*.

4.2. TEME DE CASĂ

1. Presupunem că ecuaţia ce leagă curentul de tensiunea pe o diodă este:

   $i = I_0 e^{eV/\gamma kT}$, cu e = q şi kT/q ~ 25mV.

Se cer valorile I_0 şi γ (gamma) utilizând regresia liniară asupra tabelului:

v (V)	.2	.28	.36	.44	.52	.6
ln(i)	$\ln(10^{-9})$ = -20.7232	$\ln(10^{-8})$ = -18.4207	$\ln(10^{-7})$ = -16.1181	$\ln(10^{-6})$ = -13.8155	$\ln(10^{-5})$ = -11.5129	$\ln(10^{-4})$ = -9.2103

Indicaţie:

$\ln(i/I_0) = (1/gamma)*(v/uT) \Rightarrow \ln(i) - \ln(I_0) = a*x$,
cu: $a = 1/(gamma*uT)$, iar $x = v$.

Apoi:

$\ln(i) = a*x + \ln(I_0)$,

ceea ce respectă definiţia unei regresii liniare, de formă y = ax+b.

Din aplicarea regresiei liniare asupra tabelului vor rezulta chiar a şi b. Din valoarea lui a se deduce gamma, iar din valoarea lui b rezultă I_0.

Obs.:

Valorile de tip ordonată din tabel sunt de fapt valori numerice (*scalari*), anume funcţia logaritm natural aplicată valorilor curentului din tabelul din problema interpolării. Această operaţie este necesară datorită logaritmării aplicate relaţiei curentului, în deducerea teoretică. Se vor putea, astfel, scădea valori de aceeaşi natură, în expresia erorii pătratice.

BIBLIOGRAFIE

1. I. Rusu, "*Metode numerice în electronică. Aplicaţii în limbaj C*", Editura Tehnică, Bucureşti, 1997.