CAON:
O COLECȚIE DE APLICAȚII DE OPTIMIZARE NELINIARĂ ÎN LIMBAJUL GAMS

Neculai Andrei
Research Institute for Informatics,
Center for Advanced Modeling and Optimization,
8-10, Averescu Avenue, Bucharest 1, Romania,
Academy of Romanian Scientists
E-mail: nandrei@ici.ro

În acest raport tehnic prezentăm un număr de 25 modele de optimizare neliniară, exprimate în limbajul GAMS [Brooke, *et all.*, 1998], împreună cu performanțele algoritmilor de optimizare CONOPT, KNITRO şi MINOS care fac parte integrantă din tehnologia GAMS. Scopul acestei colecții este de a asambla o multitudine de modele de optimizare neliniară care, eventual cu modificări minimale, se pot utiliza în diverse situații practice concrete. În acelaşi timp s-a avut în vedere ilustrarea capabilităților limbajului GAMS în ceea ce priveşte exprimarea diferitelor probleme ca probleme de optimizare neliniară.

Fiecare model (problemă) este introdus prin o scurtă descriere, prezentarea expresiei algebrice a acestuia, prezentarea formei modelului în limbajul GAMS, anumite elemente privind afişarea soluției, precum şi performanțele algoritmilor incluşi în tehnologia GAMS: CONOPT, KNITRO şi MINOS. Expresia GAMS a tuturor acestor aplicații se găseşte pe CD-ul asociat acestui raport tehnic. De aceea nu am mai prezentat soluția problemelor. Acolo unde a fost cazul am prezentat doar reprezentarea grafică a acestora.

Caracteristic acestor aplicații de optimizare neliniară, din această colecție, este faptul că pentru fiecare dintre acestea se prezintă semnificația fizică a variabilelor, restricțiilor, a funcției obiectiv şi a parametrilor (acolo unde este cazul). Aceasta permite utilizatorului înțelegerea procedurilor de utilizare a limbajului GAMS în ceea ce priveşte exprimarea anumitor condiții asupra variabilelor şi restricțiilor problemei, precum şi fixarea condițiilor de optimizare în cazul celor trei pachete utuilizate în acest studiu numeric.

Pe lângă o serie de lucrări menționate în bibliografie, au fost consultate colecțiile: SPENPROB Andrei [2001], [Floudas, *et. al.,* 1999], COPS [Bondarenko, Bortz şi Moré, 1999], COPS 3.0 [Dolan, Moré, Munson, 2004], MINPACK-2 [Averick, Carter, Moré, şi Xue, 1994], etc.

♦

Lista aplicațiilor din colecția CAON

Aplicaţia G1 (*Polygon*)
Dintre toate poligoanele cu n_v *laturi şi diametrul* $d \le 1$, *să se determine poligonul de arie maximă.*
Problema este descrisă în [Dolan, Moré şi Munson, 2004, pagina 3] şi [Graham, 1975]. Dacă (r_i, θ_i) sunt coordonatele vârfurilor unui poligon, atunci expresia matematică a problemei de optimizare este:

$$\max\left(\frac{1}{2}\sum_{i=1}^{n_v-1} r_{i+1} r_i \sin(\theta_{i+1} - \theta_i)\right)$$

referitor la

$$\begin{aligned} & r_i^2 + r_j^2 - 2 r_i r_j \cos(\theta_i - \theta_j) \le 1, && 1 \le i < n_v, \quad i < j \le n_v, \\ & \theta_i \le \theta_{i+1}, && 1 \le i < n_v, \\ & \theta_i \in [0, \pi], \quad r_i \ge 0, && 1 \le i \le n_v. \end{aligned}$$

Problema este interesantă deoarece are n_v^2 restricţii inegalităţi neliniare neconvexe. Dacă $n_v \to \infty$, atunci aria maximă tinde la aria unui cerc cu diametrul egal cu 1, de valoare $\pi/4 \cong 0.7854$. Problema este descrisă şi în [Andrei, 2001, pagina 328]. Expresia GAMS a acestei probleme este prezentată în figura 1.1.

```
$ontext
Polygon
Determinarea celui mai mare poligon cu n laturi de diametru
d <=1.
$offtext

$if     set n  $set nv %n%
$if not set nv $set nv 300
set i sides /i1 * i%nv%/;
alias(i,j)

scalar pi

positive variables
  r(i)
  theta(i)
variable
  polygon_area;

equations
  obj
  distance(i,j)
  ordered(i);

* equatiile
obj.. polygon_area =E= 0.5 * sum(j(i+1),
      r(i)*r(i+1)*sin(theta(i+1)-theta(i)));
distance(i,j)$(ord(j)>ord(i)).. sqr(r(i))+sqr(r(j))-
                  2*r(i)*r(j)*cos(theta(j)-theta(i)) =L=1;
```

```
ordered(i+1).. theta(i) =L= theta(i+1);

pi = 2*arctan(inf);
r.up(i) = 1;
theta.up(i) = pi;

r.l(i) = 4*ord(i)*(card(i)+1-ord(i))/sqr(card(i)+1);
theta.l(i) = pi*ord(i)/card(i);

model polygon /all/;

$onecho >bench.opt
solvers conopt knitro minos
$offecho

polygon.optfile=1;
polygon.iterlim=50000;
option nlp=bench
option reslim = 3600;
solve polygon using nlp maximizing polygon_area;
display polygon_area.l;
* End polygon
```

Fig. 1.1. Expresia GAMS a aplicației G1 (Polygon).

Pentru diverse valori ale lui n_v performanțele algoritmilor CONOPT, KNITRO şi MINOS sunt descrise în tabelele de mai jos, unde **#iter** este numărul de iterații, **time** este timpul CPU în secunde şi **vfo** este valoarea optimă a funcției obiectiv.

Tabelul 1.1

$n_v = 100,\ n = 201$ (variabile), $m = 5050$ (restricții)

	#iter	**time**	**vfo**
CONOPT	442	20.738	0.78481114
KNITRO	49	21.490	0.71973268
MINOS	48	13.238	0.6749814

Tabelul 1.2.

$n_v = 200,\ n = 401$ (variabile), $m = 20100$ (restricții)

	#iter	**time**	**vfo**
CONOPT	1066	180.730	0.78515482
KNITRO	39	264.530	0.72685233
MINOS	217	236.164	0.7322066

Tabelul 1.3.

$n_v = 300,\ n = 601$ (variabile), $m = 45150$ (restricții)

	#iter	**time**	**vfo**
CONOPT	2319	762.438	0.78529080
KNITRO	49	3476.609	0.72686364
MINOS	282	763.039	0.68840451

Aplicația G2 (*Electron*)
Distribuția electronilor pe o sferă conductoare. Dați n_p electroni, să se determine starea de echilibru a acestora pe o sferă conductoare.

Problema, cunoscută ca problema Thomson, constă în a determina configurația de energie minimă a n_p puncte încărcate electric plasate pe o sferă conductoare. Aceasta este o problemă importantă în fizică şi chimie referitoare la determinarea unei structurii de potențial Coulomb minim corespunzătoare pozițiilor unor atomi. Dacă (x_i, y_i, z_i) sunt pozițiile celor n_p puncte (electroni), atunci energia potențială a acestora este:

$$\sum_{i=1}^{n_p-1} \sum_{j=i+1}^{n_p} \left((x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2 \right)^{-\frac{1}{2}}$$

care trebuie minimizată în virtutea restricțiilor:

$$x_i^2 + y_i^2 + z_i^2 = 1, \quad i = 1, \ldots, n_p.$$

Problema este descrisă şi în [Andrei, 2001, pagina 292]. Aceasta are o multitudine de minime locale în care valoarea funcției obiectiv este apropiată de minimul global. Mai mult, numărul de minime locale creşte exponențial cu n_p. Ca atare, determinarea minimului global computațional este o problemă dificilă. Optimizatoarele profesionale determină numai punctele de minim local.
Expresia GAMS a acestei probleme este prezentată în figura 2.1.

```
$ontext
Electron
Given n electrons, find the equilibrium state distribution (of
minimal Coulomb potential) of the electrons positioned on a
conducting sphere.
$offtext

$if     set n  $set np %n%
$if not set np $set np 50

Set i       electrons /i1 * i%np%/
    ut(i,i) upper triangular part;

Alias (i,j);
    ut(i,j)$(ord(j) > ord(i)) = yes;

Variables   x(i) x-coordinate of the electron
            y(i) y-coordinate of the electron
            z(i) z-coordinate of the electron
            potential Coulomb potential;

Equations   obj     objective
            ball(i) points on unit ball;

obj.. potential =e=
```

```
  sum{ut(i,j), 1.0/sqrt(sqr(x[i]-x[j]) + sqr(y[i]-y[j]) +
                        sqr(z[i]-z[j]))};

ball(i).. sqr(x(i)) + sqr(y(i)) + sqr(z(i)) =e= 1;

* Set the starting point to a quasi-uniform distribution
* of electrons on a unit sphere

scalar pi a famous constant;
pi = 2*arctan(inf);

parameter theta(i), phi(i);
theta(i) = 2*pi*uniform(0,1);
phi(i)   = pi*uniform(0,1);

x.l(i) = cos(theta(i))*sin(phi(i));
y.l(i) = sin(theta(i))*sin(phi(i));
z.l(i) =               cos(phi(i));

model electron /all/;

electron.iterlim = 500000;
option reslim=5000;

$onecho >bench.opt
solvers conopt knitro minos
$offecho
electron.optfile=1;
electron.workfactor=20;
option nlp=bench;

solve electron using nlp minimizing potential;
* End electron
```

Fig. 2.1. Expresia GAMS a aplicației G2 (Electron).

Pentru diverse valori ale lui n_p performanțele algoritmilor CONOPT, KNITRO şi MINOS sunt descrise în tabelele de mai jos.

Tabelul 2.1

$n_p = 50$, $n = 151$ (variabile), $m = 51$ (restricții)

	#iter	time	vfo
CONOPT	37	0.602	1055.1823147
KNITRO	234	19.758	1055.1823147
MINOS	54	6.891	1055.1823147

Tabelul 2.2

$n_p = 100$, $n = 301$ (variabile), $m = 101$ (restricții)

	#iter	time	vfo
CONOPT	59	3.404	4448.4104206
KNITRO	1275	1433.160	4448.3506343
MINOS	Prea multe iterații		

Aplicația G3 (*Chain*)
Să se determine forma unui lanț de densitate uniformă, de lungime L, suspendat între două puncte fixe, cu energie potențială minimă.
Această problemă clasică, cunoscută încă ca şi curba câinelui, a fost sugerată de Hans Mittelman. Problema este descrisă în [Dolan, Moré şi Munson, 2004, pagina 9] (vezi de asemenea [Cesari, 1983], [Andrei, 2004c]).
Problema constă în a determina funcția $x(t)$, forma lanțului (înălțimea la care se află lanțul), care minimizează energia potențială:

$$\int_0^1 x(t)\left(1+x'(t)^2\right)^{1/2}\,\mathrm{d}t,$$

referitor la restricția impusă de lungimea lanțului:

$$\int_0^1 \left(1+x'(t)^2\right)^{1/2}\,\mathrm{d}t = L,$$

precum şi condițiile la capetele lanțului $x(0)=a$ şi $x(1)=b$, unde a şi b sunt înălțimile la care este suspendat lanțul la cele două capete.

O altă formulare a problemei se poate face introducând variabila de control $u(t)=x_1'(t)$ şi funcția care reprezintă energia potențială

$$x_2(t)=\int_0^t x_1(s)\left(1+u(s)^2\right)^{1/2}\,\mathrm{d}s.$$

Această formulare ne conduce la minimizarea energiei potențiale (totale) $x_2(1)$ referitor la ecuațiile

$$x_1'(t)=u,$$
$$x_2'(t)=x_1(t)\left(1+u(t)^2\right)^{1/2},$$
$$x_3'(t)=\left(1+u(t)^2\right)^{1/2}.$$

Introducând un număr n de puncte de discretizare care definesc lungimea pasului de discretizare $h=1/(n+1)$, atunci, varianta discretă a problemei este

$$\min h\sum_{i=1}^{n+1}\left(\frac{x_i+x_{i-1}}{2}\right)\sqrt{1+\left(\frac{x_i-x_{i-1}}{h}\right)^2}$$

referitor la

$$\sum_{i=1}^{n+1}\sqrt{1+\left(\frac{x_i-x_{i-1}}{h}\right)^2}=\frac{L}{h},$$

unde $x_0=a$ şi $x_{n+1}=b$ [Bondarenko, Bortz, Moré, 1999].

În reprezentarea GAMS a problemei se utilizează varianta integrării sistemului de ecuații diferențiale de mai sus. Varianta discretă de mai sus a problemei, descrisă în [Bondarenko, Bortz, Moré, 1999], a fost utilizată în pachetul SPENBAR [Andrei, 2001]. În experimentele numerice vom considera $a=1$, $b=3$ şi $L=4$.

Expresia GAMS a problemei este prezentată în figura 3.1

```
$ontext
Find the chain (of uniform density) of length L suspended
between two points with minimal potential energy.
$offtext

$if     set n  $set nh %n%
$if not set nh $set nh 1000

set nh /i0 * i%nh%/;

alias(nh,i);

scalars L  length of the suspended chain      / 4 /
        a  height of the chain at t=0 (left)  / 1 /
        b  height of the chain at t=1 (left)  / 3 /
        tf ODEs defined in [0 tf]             / 1 /
        h  uniform interval length
        n  number of subintervals
        tmin;

if (b>a, tmin = 0.25 else tmin = 0.75);
n = card(nh) - 1;
h = tf/n;

variables
  x(i)   height of the chain
  u(i)   derivative of x
  energy potential energy ;

x.fx('i0')    = a;
x.fx('i%nh%') = b;

x.l(i) = 4*abs(b-a)*((ord(i)-1)/n)*(0.5*((ord(i)-1)/n)
                      - tmin) + a;
u.l(i) = 4*abs(b-a)*(((ord(i)-1)/n) - tmin);

* Equations
equations obj, x_eqn(i), length_eqn ;

obj.. energy =e=
       0.5*h*sum(nh(i+1), x(i)*sqrt(1+sqr(u(i))) +
                          x(i+1)*sqrt(1+sqr(u(i+1))));

x_eqn(i+1).. x(i+1) =e= x(i) + 0.5*h*(u(i)+u(i+1));

length_eqn.. 0.5*h*sum(nh(i+1), sqrt(1+sqr(u(i))) +
                                sqrt(1+sqr(u(i+1)))) =e= L;

model chain /all/;

chain.optfile=1;
chain.workspace=120;
option nlp=conopt
```

```
solve chain using nlp minimizing energy;

*file res /chain.dat/;
*put res
*loop(i, put x.l(i):10:5, put/)
* End Chain
```

Fig. 3.1. Expresia GAMS a aplicației G3 (Chain).

Pentru diverse valori ale lui n_h performanțele algoritmilor CONOPT, KNITRO, MINOS şi SPENBAR sunt descrise în tabelele de mai jos.

Tabelul 3.1

$n_h = 200,\ n = 403$ (variabile), $m = 202$ (restricții)

	#iter	time	vfo
CONOPT	17	0.258	5.06891734
KNITRO	8	0.230	5.06891707
MINOS	36	1.266	5.068917

Tabelul 3.2

$n_h = 500,\ n = 1003$ (variabile), $m = 502$ (restricții)

	#iter	time	vfo
CONOPT	21	0.711	5.06857779
KNITRO	8	0.190	5.06857764
MINOS	96	10.586	5.068578
SPENBAR	3 / 20072†	117.03	5.068480

† 3 iterații majore,
20072 iterații minore în metoda Newton trunchiată.

Tabelul 3.3

$n_h = 1000,\ n = 2003$ (variabile), $m = 1002$ (restricții)

	#iter	time	Vfo
CONOPT	20	2.654	5.0685101
KNITRO	8	0.380	5.0685099
MINOS	202	73.145	5.068510

Tabelul 3.4

$n_h = 1200,\ n = 2403$ (variabile), $m = 1202$ (restricții)

	#iter	time	Vfo
CONOPT	26	4.008	5.0685019
KNITRO	8	0.490	5.0685018
MINOS	253	130.316	5.068502

Pentru $n_h = 400$ forma lanțului este arătată în figura 3.2.

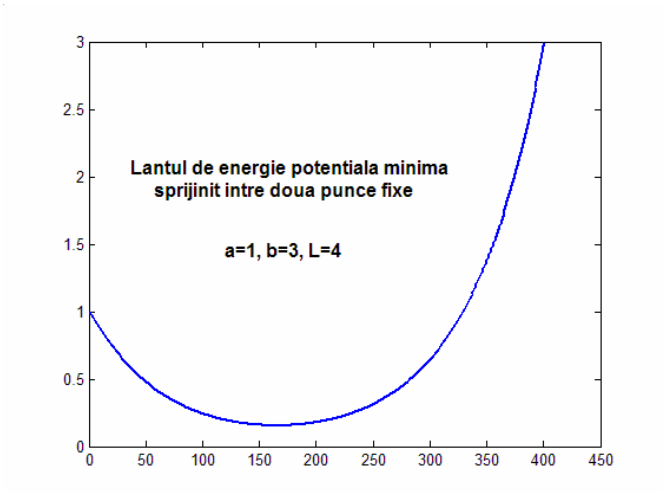

Fig. 3.2. Forma lanțului de energie potențială minimă sprijinit între două puncte fixe.

Aplicația G4 (*Rocket*)

Date masa inițială, masa combustibilului şi caracteristicile rezistenței aerului să se determine altitudinea maximă a unei rachete lansată vertical utilizând forța de tracțiune ca variabilă de control.

Astfel formulată, problema este una de control optimal. Detalii se găsesc în [Dolan, Moré şi Munson, 2004, pagina 23] (vezi de asemenea [Bryson, 1999, pp. 392-394]).

Ecuațiile de mişcare ale rachetei sunt următoarele:

$$h' = v, \qquad v' = \frac{r - D(h,v)}{m} - g(h), \qquad m' = -\frac{r}{c},$$

unde h este altitudinea de la centrul Terrei, v viteza pe verticala locului, r forța de tracțiune, D coeficientul de rezistență aerodinamică, g forța gravitațională şi c o constantă care măsoară impulsul combustibilului asupra rachetei.

Forța de tracțiune este mărginită ca:

$$0 \le r(t) \le r_{\max}.$$

Coeficientul de rezistența aerodinamică şi forța gravitațională sunt definite sub forma:

$$D(h,v) = D_c v^2 \exp\left(-h_c \frac{h - h(0)}{h(0)}\right), \qquad g(h) = g_0 \left(\frac{h(0)}{h}\right)^2,$$

unde D_c şi h_c sunt constante, iar g_0 este forța gravitațională la suprafața Terrei.

Inițial racheta se află în repaus, adică $v(0) = 0$, iar masa la sfârşitul zborului este o fracție din masa inițială, adică

$$m(t_f) = m_c m(0),$$

unde t_f este timpul final de zbor şi m_c o constantă. Pe lângă acestea, masa, altitudinea şi viteza sunt mărginite sub forma:

$$m(t_f) \le m(t) \le m(0), \qquad h(t) \ge h(0), \qquad v(t) \ge 0.$$

Prin scalare ecuațiile de mişcare ale rachetei se pot face adimensionale. În model parametrii $r_{\max}$, D_c şi c se aleg în funcție de mărimile $h(0)$, $m(0)$ şi g_0. Ca în [Bryson, 1999, pp. 392-394] definim:

$$r_{\max} = 3.5 g_0 m(0), \qquad D_c = \frac{1}{2} v_c \frac{m(0)}{g_0}, \qquad c = \frac{1}{2}\left(g_0 h(0)\right)^{1/2}.$$

Fără pierderea generalității putem presupune că $h(0) = m(0) = g_0 = 1$ şi considerăm următoarele valori pentru parametrii h_c, m_c şi v_c:

$$h_c = 500, \qquad m_c = 0.6, \qquad v_c = 620.$$

Valorile inițiale ale variabilelor modelului sunt următoarele: $t_f = 1$,

$$v(t) = \frac{t}{t_f}\left(1 - \frac{t}{t_f}\right), \qquad m(t) = (m_f - m_0)\left(\frac{t}{t_f}\right) + m_0,$$

evaluate în punctele de discretizare. Valoarea inițială a tracțiunii este $r = r_{\max}/2$.

Cu acestea, utilizând o rețea de discretizare uniformă cu n_h intervale şi schema trapezoidală de aproximare, în limbajul GAMS ecuațiile de mişcare se exprimă ca în figura 4.1.

```
$ontext
Goddard Rocket.
Maximize the final altitude of a vertically launched rocket,
using the  thrust as a control and given the initial mass, the
fuel mass, and the drag characteristics of the rocket.
$offtext

$if     set n  $set nh %n%
$if not set nh $set nh 1000

set h intervals / h0 * h%nh%/

scalars
   h_0  Initial height              / 1 /
   v_0  Initial velocity            / 0 /
   m_0  Initial mass                / 1 /
   g_0  Gravity at the surface      / 1 /
   nh   Number of intervals in mesh / %nh% /
   r_c  Thrust constant             /3.5/
   v_c  / 620 /
   h_c  / 500 /
   m_c  / 0.6 /
   D_c
   m_f  final mass
   c ;

* Constants:
c   = 0.5*sqrt(g_0*h_0);
m_f = m_c*m_0;
D_c = 0.5*v_c*(m_0/g_0);
```

```
variable   final_velocity

positive
variables  step   step size
           v(h)   velocity
           ht(h)  height
           g(h)   gravity
           m(h)   mass
           r(h)   thrust    (tractiunea)
           d(h)   drag      (rezistenta aerului) ;

* Bounds:
ht.lo(h) = h_0;

r.lo(h)  = 0.0;
r.up(h)  = r_c*(m_0*g_0);

m.lo(h)  = m_f;
m.up(h)  = m_0;

* Initial values:
ht.l(h) = 1;
v.l(h)  = ((ord(h)-1)/nh)*(1 - ((ord(h)-1)/nh));
m.l(h)  = (m_f - m_0)*((ord(h)-1)/nh) + m_0;
r.l(h)  = r.up(h)/2;
step.l  = 1/nh;

d.l(h) = D_c*sqr(v.l(h))*exp(-h_c*(ht.l(h)-h_0)/h_0);
g.l(h) = g_0*sqr(h_0/ht.l(h));

* Fixed values for variables:
ht.fx('h0')    = h_0;
v.fx('h0')     = v_0;
m.fx('h0')     = m_0;
m.fx('h%nh%') = m_f;

equations  df(h)  Drag function
           gf(h)  Gravity function
           obj
           h_eqn(h), v_eqn(h), m_eqn(h);

obj.. final_velocity =e= ht('h%nh%');

df(h).. d(h) =e= D_c*sqr(v(h))*exp(-h_c*(ht(h)-h_0)/h_0);

gf(h).. g(h) =e= g_0*sqr(h_0/ht(h));

h_eqn(h-1).. ht(h) =e= ht(h-1) + .5*step*(v(h) + v(h-1));

m_eqn(h-1).. m(h) =e= m(h-1) - .5*step*(r(h) + r(h-1))/c;

v_eqn(h-1).. v(h) =e= v(h-1)
             + .5*step*((r(h)   - D(h)   - m(h)  *g(h))/m(h)
             +(r(h-1) - D(h-1) - m(h-1)*g(h-1))/m(h-1));
```

```
model rocket /all/;

$onecho >bench.opt
  solvers conopt knitro minos
$offecho

rocket.optfile=1;
rocket.iterlim=50000;
option nlp=bench;

solve rocket using nlp maximizing final_velocity;
*file res1 /g4.dat/;
*put res1
*loop(h, put v.l(h):10:7, put/)
* End Rocket
```

Fig. 4.1. Expresia GAMS a aplicației G4 (Rocket).

Pentru diverse valori ale lui n_h performanțele algoritmilor CONOPT, KNITRO şi MINOS se prezintă în tabelele de mai jos.

Tabelul 4.1

$n_h = 500,\ \ n = 3008$ (variabile), $m = 2503$ (restricții)

	#iter	time	vfo
CONOPT	1997	22.533	1.012836662
KNITRO	70	5.938	1.012629770
MINOS	43	11.197	1.012603

Tabelul 4.2

$n_h = 1000,\ \ n = 6008$ (variabile), $m = 5003$ (restricții)

	#iter	time	vfo
CONOPT	2700	49.801	1.012835932
KNITRO	71	19.327	1.012397382
MINOS	35	38.846	1.012243

Tabelul 4.3

$n_h = 1200,\ \ n = 7208$ (variabile), $m = 6003$ (restricții)

	#iter	time	vfo
CONOPT	3175	63.117	1.01283609
KNITRO	78	29.282	1.01230050
MINOS	29	40.195	1.011832

În figurile de mai jos se arată evoluția variabilelor rachetei. Figura 4.2 arată altitudinea şi masa rachetei ca funcții de timp. Vedem că altitudinea creşte până la valoarea maximă $h = 1.01$. În acelaşi timp masa rachetei descreşte aproximativ liniar până la valoarea finală $m(t_f) = 0.6$ care este atinsă pentru $h = 186$ adică $t = 0.372$.

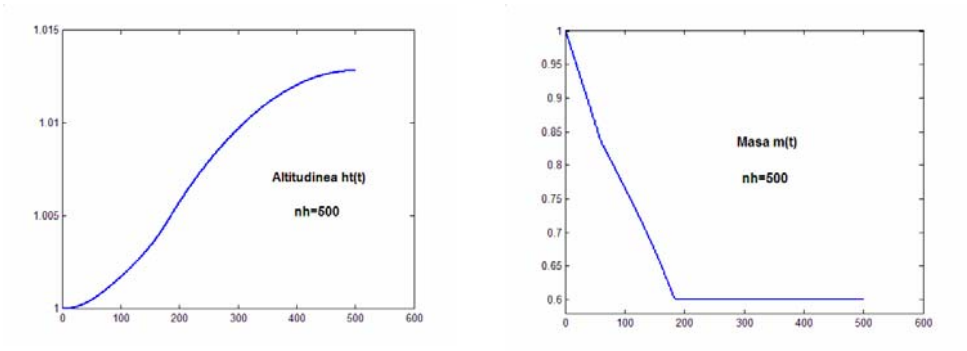

Fig. 4.2. Evoluția altitudinii $ht(t)$ şi a masei $m(t)$.

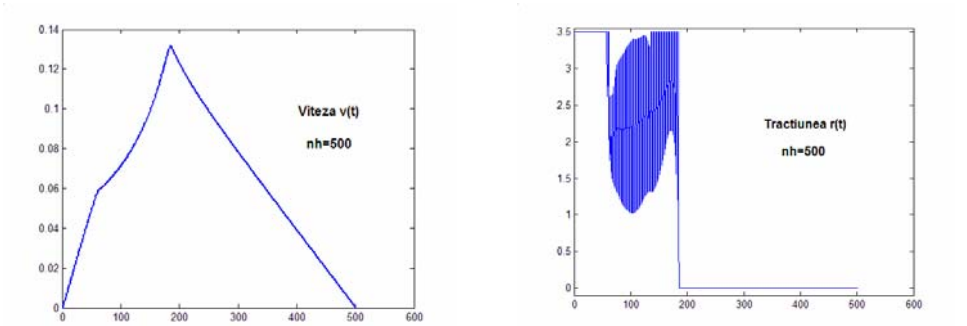

Fig. 4.3. Evoluția $v(t)$ vitezei şi a tracțiunii $r(t)$.

În figura 4.3 vedem evoluția vitezei $v(t)$ şi a tracțiunii $r(t)$ pe intervalul $[0,t_f]$, unde $t_f = 1$. Pentru $h = 185$ viteza atinge valoarea maximă $v_{\max} = 0.13218$. Pe de altă parte pe subintervalul $[59,185]$ tracțiunea are un caracter de bang-bang singular, o comportare tipică pentru sistemele de control optimal cu variabile mărginite. Este intetresant de observat cum pe subintervalul [59,185], adică pe perioada bang-bang-ului, tracțiunea are un caracter haotic, valoarea maximă a acesteia fiind limitată la 3.5 aşa cum s-a stabilit în model. De fapt, acesta este intervalul critic în lansarea oricărei rachete. Pe de altă parte, vedem cum tracțiunea are o valoare maximă de 3.5 pe durata $[0,58]$ ceea ce permite ridicarea rachetei. La momentul $h = 185$ viteza are valoarea maximă, de aici încolo tracțiunea se anulează (vezi fig. 4.3).

Este foarte instructiv să vedem comportarea tracțiunii pentru o discretizare mai fină. În figura 4.4 se arată evoluția tracțiunii pentru $nh = 1000$. Vedem imediat cum pe intervalul critic, în care apare bang-bang-ul, se detaliază evoluția tracțiunii. Caracterul haotic se menține, dar pentru un interval de timp destul de mic, către finalul intervalului de bang-bang, tracțiunea atinge valoarea maximă. Aceasta asigură obținerea vitezei maxime a rachetei. După aceasta tracțiunea se anulează până la sfârşitul intervalului de evoluție a rachetei.

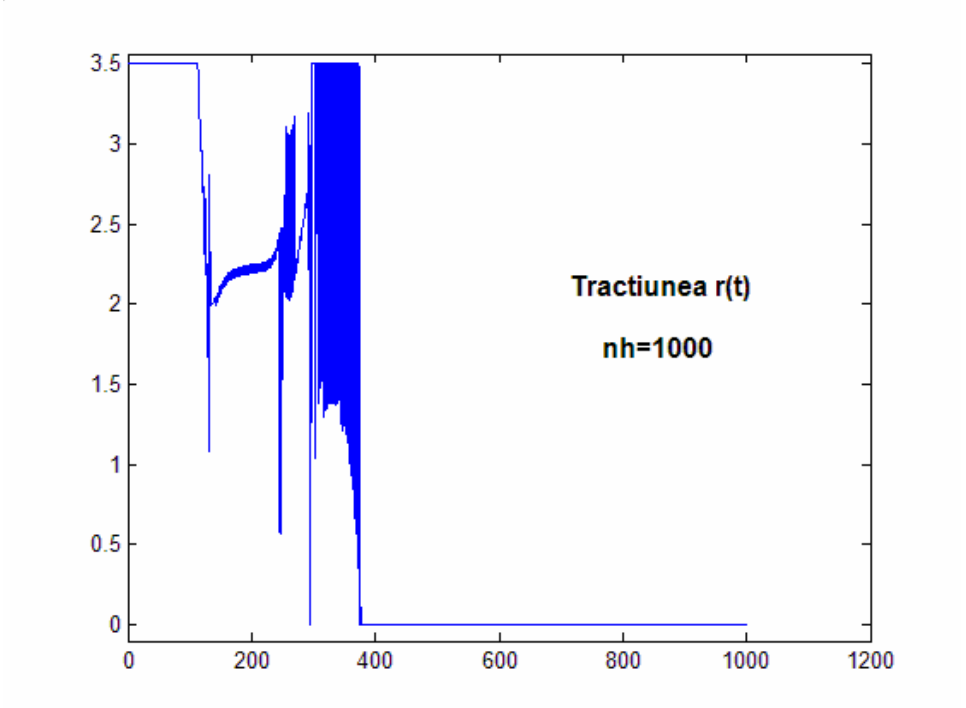

Fig. 4.4. Evoluția tracțiunii $r(t)$ pentru $nh = 1000$.

Din figura 4.4. vedem că tracțiunea are aceeaşi comportare. Pentru $h \in [0,113]$ tracțiunea are valoarea maximă de 3.5 care permite ridicarea rachetei. Pentru $h \in [114,377]$ tracțiunea are o comportare de bang-bang singular în care se vede că aceasta de multe ori atinge valoarea maximă admisă. De la momentul 378 încolo tracțiunea este nulă. Şi în acest caz vedem corelația care există între viteză şi tracțiune, pentru $h = 375$ viteza atinge valoarea maximă $v_{\max} = 0.1303938$ după care începe să scadă. De aici încolo tracțiunea este nulă.

Aplicația G5 (*Camshape*)
În această aplicație este vorba de optimizarea formei unei came. Problema a fost modelată de Anițescu şi Şerban [1998] şi constă în a *maximiza aria unei valve prin rotația unei came convexe de curbură şi rază date.*
Problema este descrisă în [Dolan, Moré şi Munson, 2004, pagina 7]. Presupunem că forma camei este circulară pentru un unghi de valoare $6\pi/5$ din circumferință, de rază $r_{\min}$. Variabilele de proiectare sunt r_i, $i = 1,\ldots,n$, care reprezintă raza camei la unghiuri egal distribuite de-a lungul unghiului $2\pi/5$. Problema constă în a maximiza aria valvei, adică a maximiza funcția

$$f(r) = \pi r_v^2 \left(\frac{1}{n} \sum_{i=1}^{n} r_i \right)$$

referitor la următoarele restricții asupra razelor r_i, $i = 1,\ldots,n$. În funcția de mai sus r_v este un parametru al problemei care depinde de geometria valvei, considerat de valoare $r_v = 1$. Asupra razelor se impun restricțiile

$$r_{\min} \le r_i \le r_{\max}.$$

Cerința de convexitate a camei este exprimată sub forma

$$\text{aria}(r_{i-1}, r_{i+1}) \le \text{aria}(r_{i-1}, r_i) + \text{aria}(r_i, r_{i+1}),$$

unde $\text{aria}(r_i, r_j)$ este aria triunghiului determinat de origine şi punctele r_i şi r_j de pe suprafața camei. Această relație de convexitate se poate încă exprima sub forma

$$2r_{i-1}r_{i+1}\cos(\theta) \le r_i(r_{i-1} + r_{i+1}), \quad i = 0,\ldots,n+1,$$

unde $r_{-1} = r_0 = r_{\min}$, $r_{n+1} = r_{\max}$, $r_{n+2} = r_n$ şi $\theta = 2\pi/5(n+1)$. Cerința de curbură se exprimă sub forma

$$-\alpha \le \left(\frac{r_{i+1} - r_i}{\theta}\right) \le \alpha, \quad i = 0,\ldots,n.$$

În această aplicație considerăm $r_{\min} = 1$, $r_{\max} = 2$ şi $\alpha = 1.5$ în restricția de curbură. Deoarece forma camei este simetrică, vom considera numai o jumătate din unghiul de proiectare.

Expresia GAMS a acestei probleme este arătată în figura 5.1.

```
$ontext
Maximize the area of the valve opening for one rotation of a
convex cam with constraints on the curvature and on the radius
of the cam.
$offtext

$if not set n $set n 800

Set i discretization points /i1 * i%n%/;

Alias (i,j);

Scalar
    R_v     design parameter related to the valve shape /1/
    R_max   maximum allowed radius of the cam           /2/
    R_min   minimum allowed radius of the cam           /1/
    pi      a famous constant
    alpha   curvature limit parameter                   /1.5/
    d_theta angle between discretization points;

pi = 2*arctan(inf);
d_theta = 2*pi/(5*(%n%+1));

set first(i), last(i), middle(i);
first('i1')  = yes;
last('i%n%') = yes;
middle(i) = yes; middle(first) = no; middle(last) = no;

Variables  r(i)      radius of the cam at discretization points
           rdiff(i)  intermediate
```

```
            area      valve area;

* Bounds
r.lo(i) = R_min;
r.up(i) = R_max;

rdiff.lo(i(j+1)) = -alpha*d_theta;
rdiff.up(i(j+1)) =  alpha*d_theta;

r.lo('i1') = max(-alpha*d_theta + R_min, r.lo('i1'));
r.up('i1') = min( alpha*d_theta + R_min, r.up('i1'));

r.lo('i%n%') =  max(R_max - alpha*d_theta, r.lo('i%n%'));
r.up('i%n%') =  min(R_max + alpha*d_theta, r.up('i%n%'));

r.up('i1') = min( R_min/(2*cos(d_theta)-1), r.up('i1'));

* Initial values
r.l(i) = (R_min+R_max)/2;

Equations  obj              objective
           convexity(i)
           convex_edge1(i)
           convex_edge3(i)
           convex_edge4(i)
           eqrdiff(i);

obj.. area =e= ((pi*R_v)/%n%) * sum(i, r(i));

convexity(middle(i)).. -r(i-1)*r(i) - r(i)*r(i+1) +
                       2*r(i-1)*r(i+1)*cos(d_theta) =l= 0;

convex_edge1(first(i)).. -R_min*r(i) - r(i)*r(i+1) +
                       2*R_min*r(i+1)*cos(d_theta) =l= 0;

convex_edge3(last(i)).. -r(i-1)*r(i) - r(i)*R_max +
                       2*r(i-1)*R_max*cos(d_theta) =l= 0;

convex_edge4(last(i)).. -2*R_max*r(i) +
                       2*sqr(r(i))*cos(d_theta) =l= 0;

eqrdiff(j(i+1)).. rdiff(i) =e= r(i+1) - r(i);

model camshape /all/;

*$onecho >bench.opt
*  solvers conopt
*$offecho

camshape.optfile=1;
camshape.iterlim=50000;
camshape.workspace=200;

option nlp=minos;

solve camshape using nlp maximizing area;
```

```
*file rez /camshape.dat/;
*put rez
*loop(i, put r.l(i):10:5, put/)
* End camshape
```

Fig. 5.1. Expresia GAMS a aplicației G5 (Camshape).

În tabelele de mai jos se prezintă performanțele algoritmilor CONOPT şi KNITRO.

Tabelul 5.1

$n = 800$, $n = 1600$ (variabile), $m = 1601$ (restricții)

	#iter	time	vfo
CONOPT	44	0.750	4.27427414
KNITRO	47	0.911	4.27401472

Tabelul 5.2

$n = 1000$, $n = 2000$ (variabile), $m = 2001$ (restricții)

	#iter	time	vfo
CONOPT	66	1.251	4.27399125
KNITRO	49	1.211	4.27366794

Tabelul 5.3

$n = 10000$, $n = 20000$ (variabile), $m = 20001$ (restricții)

	#iter	time	vfo
CONOPT	46	22.673	4.27265943
KNITRO	54	32.136	4.26883424

În figura 5.2 se arată forma camei în domeniul de unghi $2\pi/5$, pentru $\alpha = 1.5$.

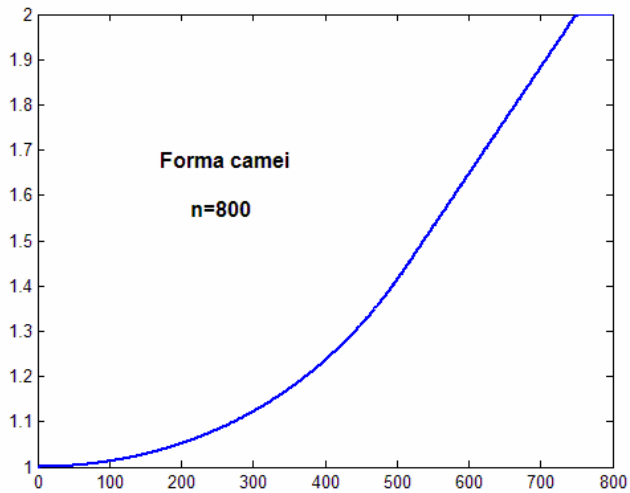

Fig. 5.2. Forma camei pentru $\alpha = 1.5$.

Aplicația G6 (*Robot*)
Minimizarea timpului de deplasare al unui robot între două puncte fixe.
Problema a fost formulată de Mössner-Beigel [1995] în teza sa de diplomă. Este vorba despre un braț robotizat format dintr-o bară rigidă de lungime L care poate parcurge o distanță ρ de la origine. Dacă punctul de articulație al brațului este originea unui sistem sferic de coordonate, atunci problema se poate exprima în termenii distanței ρ, a unghiului din planul orizontal θ, a unghiului din planul vertical φ, a mărimilor de control $(u_\rho, u_\theta, u_\varphi)$ asociate acestor elemente şi a timpului final t_f.
Asupra acestor variabile se impun condițiile:

$$\rho(t) \in [0, L], \quad |\theta(t)| \le \pi, \quad 0 \le \varphi(t) \le \pi,$$
$$|u_\rho| \le 1, \quad |u_\theta| \le 1, \quad |u_\varphi| \le 1.$$

Ecuațiile de mişcare ale robotului sunt:

$$L\rho'' = u_\rho, \quad I_\theta \theta'' = u_\theta, \quad I_\varphi \varphi'' = u_\varphi,$$

unde I este momentul de inerție, definit ca:

$$I_\theta = \frac{((L-\rho)^3 + \rho^3)}{3} \sin(\varphi)^2, \qquad I_\varphi = \frac{(L-\rho)^3 + \rho^3}{3}.$$

Pentru acest sistem de ecuații diferențiale se impun condițiile la limită:

$$\rho(0) = \rho(t_f) = 4.5, \qquad \theta(0) = 0, \ \theta(t_f) = \frac{2\pi}{3}, \qquad \varphi(0) = \varphi(t_f) = \frac{\pi}{4},$$
$$\rho'(0) = \theta'(0) = \varphi'(0) = \rho'(t_f) = \theta'(t_f) = \varphi'(t_f) = 0.$$

Valorile inițiale ale variabilelor ρ şi φ sunt $\rho = 4.5$, respectiv $\varphi = \pi/4$ pentru toate punctele de discretizare. Similar, valorile inițiale ale variabilei θ sunt date de valorile funcției

$$\theta(t) = \frac{2\pi}{3}\left(\frac{t}{t_f}\right)^2$$

calculate în punctele de discretizare. Pentru t_f ca valoare inițială se consideră $t_f = 1$. Valorile inițiale ale tuturor mărimilor de comandă sunt nule.
Deoarece sistemul de coordonate sferice nu este un sistem inerțial, în această formulare, modelul matematic al robotului nu conține forțele Coriolis şi centrifugale.
După cum vedem ecuațiile de mişcare ale robotului sunt un sisteme de ecuații diferențiale de ordinul doi. Introducând variabilele suplimentare ρ', θ' şi φ', atunci acestea se exprimă ca un sistem de ecuații diferențiale de ordinul unu. Discretizând intervalul $[0, t_f]$ uniform în n_h intervale şi utilizând schema de aproximare trapezoidală, expresia GAMS a modelului de minimizare a timpului de deplasare este ca în figura 6.1.

```
$ontext
Minimize the time taken for a robot arm to travel between two
points.
$offtext

$if     set n  $set nh %n%
$if not set nh $set nh 200

sets  h intervals / h0 * h%nh%/

scalars
    pi a famous constant
    nh number of intervals / %nh% /
    L  total length of arm / 5 /
    max_u_rho  / 1 /
    max_u_the  / 1 /
    max_u_phi  / 1 /;

pi = 2*arctan(inf);

variables
    rho(h)        distance from the origin
    the(h)        horizontal angle
    phi(h)        vertical angle
    rho_dot(h)
    the_dot(h)
    phi_dot(h)
    u_rho(h)      control
    u_the(h)
    u_phi(h)
    step
    tf            final time
    i_the(h)      moment of inertia
    i_phi(h) ;

* Bounds
rho.lo(h) = 0;    rho.up(h) = L;
the.lo(h) = -pi;  the.up(h) = pi;
phi.lo(h) = 0;    phi.up(h) = pi;

u_rho.lo(h) = -max_u_rho;  u_rho.up(h) = max_u_rho;
u_the.lo(h) = -max_u_the;  u_the.up(h) = max_u_the;
u_phi.lo(h) = -max_u_phi;  u_phi.up(h) = max_u_phi;

i_the.lo(h) = 0.0001;
i_phi.lo(h) = 0.0001;

set firstlast(h) / h0, h%nh% /;

* Fixed variables
the.fx('h0')    = 0;
the.fx('h%nh%') = 2*pi/3;

rho.fx(firstlast)      = 4.5;
phi.fx(firstlast)      = pi/4;
rho_dot.fx(firstlast)  = 0;
```

```
the_dot.fx(firstlast)  = 0;
phi_dot.fx(firstlast)  = 0;
i_phi.fx(firstlast(h)) = (power(L-rho.l(h),3)+
                          power(rho.l(h),3))/3.0;
i_the.fx(firstlast(h)) = i_phi.l(h)*sqr(sin(phi.l(h)));

*Initialization
rho.l(h) = 4.5;
the.l(h) = (2*pi/3)*sqr(ord(h)/nh);
phi.l(h) = pi/4;

rho_dot.l(h) = 0.0;
the_dot.l(h) = (4*pi/3)*(ord(h)/nh);
phi_dot.l(h) = 0.0;

step.l = 1/nh;

i_phi.l(h) = (power(L-rho.l(h),3)+power(rho.l(h),3))/3.0;
i_the.l(h) = i_phi.l(h)*sqr(sin(phi.l(h)));

equations
     tf_eqn
     rho_eqn(h)
     the_eqn(h)
     phi_eqn(h)
     u_rho_eqn(h)
     u_the_eqn(h)
     u_phi_eqn(h)
     i_the_eqn(h)
     i_phi_eqn(h);

tf_eqn.. tf =e= step*nh;

i_phi_eqn(h).. i_phi(h) =e= (power(L-rho(h),3)+
                             power(rho(h),3))/3.0;

i_the_eqn(h).. i_the(h) =e= i_phi(h)*sqr(sin(phi(h)));

rho_eqn(h-1)..  rho(h) =e= rho(h-1) + 0.5*step*(rho_dot(h) +
                           rho_dot(h-1));

the_eqn(h-1)..  the(h) =e= the(h-1) + 0.5*step*(the_dot(h) +
                           the_dot(h-1));
phi_eqn(h-1)..  phi(h) =e= phi(h-1) + 0.5*step*(phi_dot(h) +
                           phi_dot(h-1));

u_rho_eqn(h-1)..  rho_dot(h) =e=
                    rho_dot(h-1) + 0.5*step*(u_rho(h) +
                    u_rho(h-1))/L;

u_the_eqn(h-1)..  the_dot(h) =e=
          the_dot(h-1) + 0.5*step*(u_the(h)/i_the(h) +
          u_the(h-1)/i_the(h-1));

u_phi_eqn(h-1)..  phi_dot(h) =e=
          phi_dot(h-1) + 0.5*step*(u_phi(h)/i_phi(h) +
```

```
          u_phi(h-1)/i_phi(h-1));

model robot /all/;

$onecho >bench.opt
  solvers conopt
$offecho

robot.optfile=1;
robot.iterlim=50000;
robot.workspace=120;

option nlp=bench

solve robot miniziming tf using nlp;

*file res /robot.dat/;
*put res
*loop(h, put rho.l(h):10:5, put/)
*End Robot
```

Fig. 6.1. Expresia GAMS a aplicației G6 (Robot).

În tabelul 6.1 se prezintă performanțele algoritmului CONOPT pentru diferite valori ale lui n_h, unde n este numărul de variabile al problemei, m numărul de restricții, iar **time** este exprimat în secunde. Din acest tabel se vede că, aparent pentru orice discretizare, valoarea optimă a funcției obiectiv este aceeaşi.

Tabelul 6.1.
Performanțele algoritmului CONOPT pentru rezolvarea aplicației G6.

n_h	n	m	**#iter**	**time**	**vfo**
100	1113	803	200	0.871	9.1426853843
200	2213	1603	279	1.863	9.1413954984
400	4413	3203	429	7.379	9.1410260259
800	8813	6403	786	18.598	9.1409409706
1000	11013	8003	953	27.520	9.1409312779
1200	13213	9603	996	36.152	9.1409253934
1400	15413	11203	1302	53.035	9.1409214713
2000	22013	16003	1820	110.168	9.1409165109
2100	23113	16803	1640	115.055	9.1409158906
2200	24213	17603	1713	123.238	9.1409158061
2400	26413	19203	2160	154.484	9.1409151373
2600	28613	20803	2336	188.238	9.1409145137
2800	30813	22403	2506	213.398	9.1409139367
3000	33013	24003	2274	225.953	9.1409137707

Problemă greu de rezolvate pentru pachetele KNITRO şi MINOS. KNITRO cere un număr de iterații prea mare. MINOS are dificultăți în evaluarea funcțiilor problemei. În figurile de mai jos se prezintă evoluția variabilelor asociate robotului.

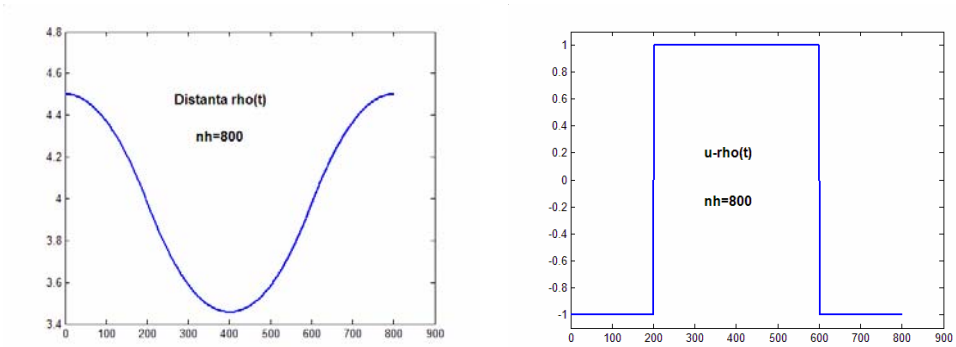

Fig. 6.2. Evoluția variabilelor $\rho(t)$ şi $u_\rho(t)$.

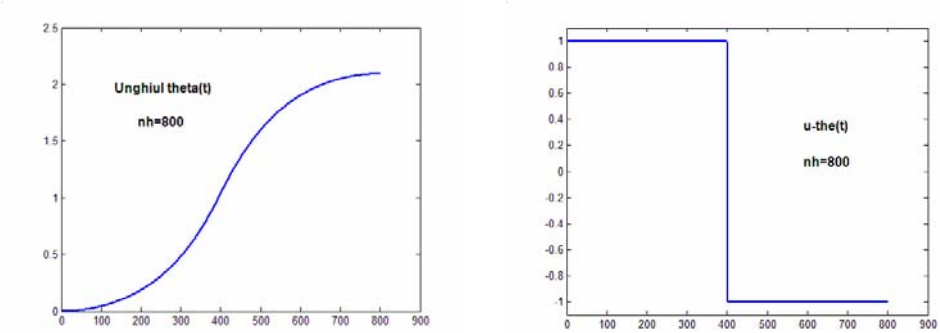

Fig. 6.3. Evoluția variabilelor $\theta(t)$ şi $u_\theta(t)$.

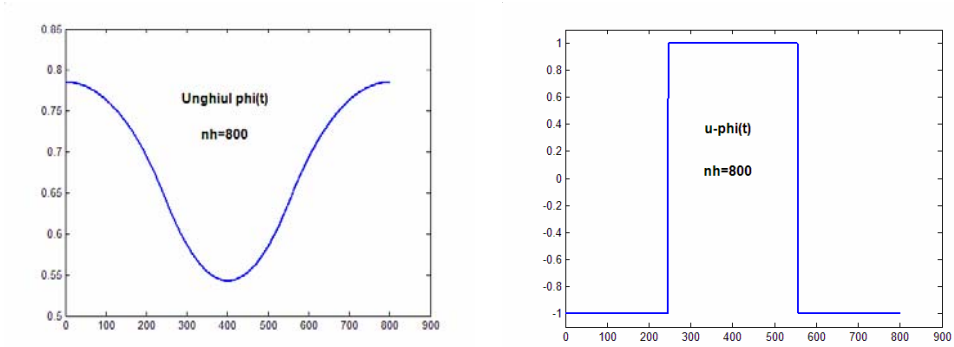

Fig. 6.4. Evoluția variabilelor $\varphi(t)$ şi $u_\varphi(t)$.

Problema este de a minimiza timpul de deplasare ale brațului robotului între două puncte fixe. Evoluția variabilelor $\rho(t)$ (distanța), $\theta(t)$ (unghiul din planul orizontal) şi $\varphi(t)$ (unghiul din planul vertical), în raport cu condițiile inițiale, este foarte naturală. Şi în acest caz, variabilele fiind mărginite, mărimile de control asociate acestor variabile au un caracter de bang-bang.

Aplicația G7 (*Steering*)
Minimizarea timpului de deplasare a unei particule, supusă unei tracțiuni date, pentru a realiza o altitudine şi viteză finală impuse.
Utilizând [Bryson şi Ho, 1975, pp. 59-62] şi [Dolan, Moré şi Munson, 2004, pp. 21-22], ecuațiile de mişcare sunt

$$\ddot{y}_1 = a\cos u, \qquad \ddot{y}_2 = a\sin u,$$

unde (y_1, y_2) este poziția particulei, u este unghiul de control cu

$$|u(t)| \le \frac{\pi}{4},$$

iar a este forța de tracțiune. Inițial particula este în repaus, astfel încât

$$y_1(0) = y_2(0) = \dot{y}_1(0) = \dot{y}_2(0) = 0.$$

Problema este de a minimiza timpul final de deplasare a particulei t_f astfel încât aceasta atinge o altitudine dată $y_2(t_f)$ şi o viteză finală cunoscută $(\dot{y}_1(t_f), \dot{y}_2(t_f))$. În această aplicație considerăm: $a = 100$ şi $y_2(t_f) = 5$, $\dot{y}_1(t_f) = 45$ şi $\dot{y}_2(t_f) = 0$. Utilizând o discretizare uniformă a timpului de deplasare şi schema trapezoidală de aproximare pe n_h intervale expresia GAMS a problemei este ca în figura 7.1.

```
$ontext
Minimize the time take for a particle, acted upon by a thrust
of constant magnitude, to achieve a given altitude and terminal
velocity.
$offtext

$if set n $set nh %n%
$if not set nh $set nh 2000

sets h intervals / h0 * h%nh% /
c coordinates /
      y1 first position coordinate
      y2 second position coordinate
      y3 first velocity coordinate
      y4 second velocity coordinate /

scalars pi a famous constant

nh number of intervals / %nh% /

a magnitude of force / 100.0 / ;

variables u(h)            control
          y(c,h)          coordinates
          tf              final time ;

positive variables step   step size ;

y.l('y2',h) = 5*(ord(h)-1)/nh;
y.l('y3',h) = 45*(ord(h)-1)/nh;
step.l = 1.0/nh;
```

```
equations tf_eqn, pos_eqn(c,h), velo1_eqn(h), velo2_eqn(h);

tf_eqn.. tf =e= step*nh;

pos_eqn(c+2,h+1).. y(c,h+1) =e= y(c,h) +
                      0.5*step*(y(c+2,h) + y(c+2,h+1));

velo1_eqn(h+1).. y('y3',h+1) =e= y('y3',h) +
                      0.5*step*(a*cos(u(h)) + a*cos(u(h+1)));

velo2_eqn(h+1).. y('y4',h+1) =e= y('y4',h) +
                      0.5*step*(a*sin(u(h)) + a*sin(u(h+1)));

pi = 2*arctan(inf);

u.lo(h) = -pi/2;
u.up(h) = pi/2;

y.fx(c,'h0') = 0;
y.fx('y2','h%nh%') = 5;
y.fx('y3','h%nh%') = 45;
y.fx('y4','h%nh%') = 0;

model steering /all/;

$onecho >bench.opt
  solvers conopt knitro
$offecho
steering.optfile=1;

option nlp=bench;

solve steering using nlp minimizing tf;

*End steering
```

Fig. 7.1. Expresia GAMS a aplicației G7 (Steering).

În tabelele de mai jos se arată performanțele algoritmilor CONOPT, KNITRO şi MINOS pentru diferite valori ale lui n_h considerând că valorile inițiale ale lui $y_2(t)$ şi $y_3(t) = \dot{y}_1(t)$ sunt alese ca funcții de timpul t de forma:

$$y_1(t) = 5\left(\frac{t}{t_f}\right), \quad y_3(t) = 45\left(\frac{t}{t_f}\right).$$

Tabelul 7.1

$n_h = 400$, $n = 2007$ (variabile), $m = 1601$ (restricții)

	#iter	time	vfo
CONOPT	875	3.695	0.5545724137
KNITRO	18	0.510	0.5545724136
MINOS	57	7.551	0.554572

Tabelul 7.2

$n_h = 800,\ n = 4007$ (variabile), $m = 3201$ (restricții)

	#iter	time	vfo
CONOPT	130	13.809	0.5545712623
KNITRO	22	1.271	0.5545712622

Tabelul 7.3

$n_h = 1000,\ n = 5007$ (variabile), $m = 4001$ (restricții)

	#iter	time	vfo
CONOPT	27	3.508	0.5545712366
KNITRO	22	1.882	0.5545711240

Tabelul 7.4

$n_h = 2000,\ n = 10007$ (variabile), $m = 8001$ (restricții)

	#iter	time	vfo
CONOPT	83	25.406	0.5545709399
KNITRO	30	7.020	0.5545709433

Aplicația G8 (*Planor*)

Să se maximizeze poziția finală orizontală a unui planor suspendat supus unei forțe ascensionale termice date.

Ecuațiile de mişcare ale planorului suspendat sunt [Bulirsch, *et al*, 1993]:

$$x'' = \frac{1}{m}\left(-L\sin\eta - D\cos\eta\right), \qquad y'' = \frac{1}{m}\left(L\cos\eta - D\sin\eta\right) - g,$$

unde (x, y) este poziția planorului, m masa acestuia, g accelerația gravitațională şi funcția η este definită de:

$$\sin\eta = \frac{w(x,y')}{v(x,x',y')}, \qquad \cos\eta = \frac{x'}{v(x,x',y')},$$

unde

$$v(x,x',y') = \sqrt{x'^2 + w(x,y')^2}, \qquad w(x,y') = y' - u(x),$$

$$u(x) = u_c\left(1 - r(x)\right)\exp\left(-r(x)\right), \qquad r(x) = \left(\frac{x}{r_c} - 2.5\right)^2,$$

în care constantele u_c şi r_c au valorile $u_c = 2.5$ şi respectiv $r_c = 100$. Vedem că forța ascensională u este pozitivă în vecinătatea lui $x = 2.5r_c$, dar se reduce la zero exponențial departe de $x = 2.5r_c$. Funcțiile D şi L sunt definite ca:

$$D(x,x',y',c_L) = \frac{1}{2}\left(c_0 + c_1 c_L^2\right)\rho S v(x,x',y')^2, \quad L(x,x',y',c_L) = \frac{1}{2}c_L \rho S v(x,x',y')^2,$$

unde S este aria aripei, ρ densitatea aerului, c_L coeficientul de ridicare aerodinamică şi $c_0 + c_1 c_L^2$ este coeficientul de rezistenţa a aerului. Pentru acest planor considerăm:

$$c_0 = 0.034, \quad c_1 = 0.069662, \quad S = 14, \quad \rho = 1.13.$$

Coeficientul de ridicare aerodinamică c_L satisface restricţiile de mărginire:

$$0 \le c_L \le c_{\max}.$$

În acelaşi timp se impun condiţiile $x \ge 0$ şi $x' \ge 0$. În această problemă considerăm $c_{\max} = 1.4$, $m = 100$, $g = 9.81$ şi condiţiile la limită:

$$x(0) = 0, \quad y(0) = 1000, \quad y(t_f) = 900,$$

$$x'(0) = x'(t_f) = 13.23, \quad y'(0) = y'(t_f) = -1.288.$$

Pentru rezolvarea acestei probleme intervalul $[0, t_f]$ se discretizează uniform în n_h intervale, iar ecuaţiile de mişcare sunt aproximate prin schema trapezoidală. Condiţiile iniţiale sunt

$$x(t) = x(0) + x'(0)\left(\frac{t}{t_f}\right), \quad y(t) = y(0) + (y(t_f) - y(0))\left(\frac{t}{t_f}\right), \quad c_L(t) = c_{\max}/2.$$

evaluate în punctele de discretizare. Se consideră $t_f = 1$.

Expresia GAMS a modelului discretizat cu aceste condiţii (iniţiale şi la limită) este prezentat în figura 8.1.

```
$ontext
Maximize the final horizontal position of a hang glider while
in the presence of a thermal updraft.
$offtext

$if set n $set nh %n%
$if not set nh $set nh 1000
sets c coordinates / x distance
                     y altitude /
h intervals        / h0 * h%nh%/ ;

alias(h,i);

scalars nh   Number of intervals in mesh  / %nh% /
cL_min       bound on control variable    / 0.0 /
cL_max       bound on control variable    / 1.4 /
u_c   / 2.5 /
r_0   / 100 /
m     / 100 /
g     / 9.81 /
c0    / 0.034 /
c1    / 0.069662 /
S     / 14 /
rho   / 1.13 / ;

parameters c_0(c) initial position  / x 0, y 1000/
           v_0(c) initial velocity  / x 13.23, y -1.288/
```

```
            c_f(c) final position    / y 900/
            v_f(c) final velocity    / x 13.23, y -1.288/ ;

variables t_f
          pos(c,h)   position x distance y altitude
          vel(c,h)   velocity x distance y altitude
          cl(h)      control variables
          r(h)       the r function
          u(h)       the u function
          w(h)       the w function
          v(h)       the v function
          D(h)       the D function
          L(h)       the L function
          v_dot(c,h)
          final_x
          step       step size

positive variables step;

equations tf_eqn
          rdef(h)
          udef(h)
          wdef(h)
          vdef(h)
          Ddef(h)
          Ldef(h)
          vx_dot_def(h)
          vy_dot_def(h)
          obj
          pos_eqn(c,h)
          vel_eqn(c,h);

tf_eqn.. t_f =e= step*nh;

rdef(i).. r[i] =e= sqr(pos['x',i]/r_0 - 2.5);

udef(i).. u[i] =e= u_c*(1-r[i])*exp(-r[i]);

wdef(i).. w[i] =e= vel['y',i] - u[i];

vdef(i).. v[i] =e= sqrt(sqr(vel['x',i]) + sqr(w[i]));

Ddef(i).. D[i] =e= .5*(c0+c1*sqr(cL[i]))*rho*S*sqr(v[i]);

Ldef(i).. L[i] =e= .5* cL[i] *rho*S*sqr(v[i]);

vx_dot_def(i).. v_dot['x',i] =e= (-L[i]*w[i]/v[i] -
                                  D[i]*vel['x',i]/v[i])/m;

vy_dot_def(i).. v_dot['y',i] =e= ( L[i]*vel['x',i]/v[i] -
                                  D[i]*w[i]/v[i])/m - g;

obj.. final_x =e= pos('x','h%nh%');

pos_eqn(c,i-1).. pos[c,i] =e= pos[c,i-1] +
                        .5*step*(vel[c,i] + vel[c,i-1]);
```

```
vel_eqn(c,i-1).. vel[c,i] =e= vel[c,i-1] +
                        .5*step*(v_dot[c,i] + v_dot[c,i-1]);

* Boundary Conditions
cl.lo(h) = cL_min;
cl.up(h) = cL_max;
pos.lo('x',h) = 0;
vel.lo('x',h) = 0;
v.lo(h) = 0.01;

* Fixed values
pos.fx(c,'h0') = c_0(c);
pos.fx('y','h%nh%') = c_f('y');
vel.fx(c,'h0') = v_0(c);
vel.fx(c,'h%nh%') = v_f(c);

* Initial point
pos.l('x',h) = c_0('x') + v_0('x')*((ord(h)-1)/nh);
pos.l('y',h) = c_0('y') +
               ((ord(h)-1)/nh)*(c_f('y') - c_0('y'));
vel.l(c,h) = v_0(c);
cL.l(h) = cL_max/2;
step.l = 1.0/nh;

* Initial values for intermediate variables
t_f.l = step.l*nh;
r.l[i] = sqr(pos.l['x',i]/r_0 - 2.5);
u.l[i] = u_c*(1-r.l[i])*exp(-r.l[i]);
w.l[i] = vel.l['y',i] - u.l[i];
v.l[i] = sqrt(sqr(vel.l['x',i]) + sqr(w.l[i]));
D.l[i] = .5*(c0+c1*sqr(cL.l[i]))*rho*S*sqr(v.l[i]);
L.l[i] = .5* cL.l[i] *rho*S*sqr(v.l[i]);
v_dot.l['x',i] = (-L.l[i]*w.l[i]/v.l[i] -
                 D.l[i]*vel.l['x',i]/v.l[i])/m;
v_dot.l['y',i] = ( L.l[i]*vel.l['x',i]/v.l[i] -
                 D.l[i]*w.l[i]/v.l[i])/m - g;

model planor /all/;

$onecho >bench.opt
  solvers conopt knitro
$offecho
planor.optfile=1;

option nlp=bench;

planor.workspace = 40;

solve planor maximizing final_x using nlp;

*file res /g8.dat/;
*put res
*loop(h, put cl.l(h):10:7, put/)
* End planor
```

Fig. 8.1. Expresia GAMS a aplicației G8 (Planor).

În tabelele de mai jos se prezintă performanțele algoritmilor CONOPT şi KNITRO.

Tabelul 8.1

$n_h = 200,\ n = 2616$ (variabile), $m = 2410$ (restricții)

	#iter	Time	vfo
CONOPT	704	23.781	1248.8094263
KNITRO	366	21.671	1248.8963550

Tabelul 8.2

$n_h = 400,\ n = 5216$ (variabile), $m = 4810$ (restricții)

	#iter	Time	vfo
CONOPT	1078	45.328	1247.9743175
KNITRO	508	86.244	1247.9743047

Tabelul 8.3

$n_h = 800,\ n = 10416$ (variabile), $m = 9610$ (restricții)

	#iter	Time	vfo
CONOPT	1357	176.484	1247.9838908
KNITRO	599	309.264	1247.9838658

Tabelul 8.4

$n_h = 1000,\ n = 13016$ (variabile), $m = 12010$ (restricții)

	#iter	Time	vfo
CONOPT	1320	177.633	1247.9852155
KNITRO	662	498.276	1247.9851843

Tabelul 8.5

$n_h = 1200,\ n = 15616$ (variabile), $m = 14410$ (restricții)

	#iter	Time	vfo
CONOPT	1547	217.242	1247.9859370
KNITRO	604	631.468	1247.9858984

În figurile de mai jos se prezintă evoluția anumitor variabile ale acestei probleme. Figura 8.2 arată evoluția în timp a altitudinii $y(t)$ şi a coeficientului de ridicare ascensională $c_L(t)$. Figura 8.3 prezintă evoluția în timp a vitezelor $x'(t)$ şi $y'(t)$. Din figura 8.2 vedem că planorul începe zborul de la altitudinea $y(0) = 1000$. Apoi coboară până când acesta întâlneşte curentul ascendent centrat în jurul valorii $x = 2.5r_c = 250$. Ca atare, acesta urcă şi apoi coboară până la altitudinea finală $y(t_f) = 900$. În ceea ce priveşte evoluția vitezelor, din figura 8.3 vedem comportarea de tip bang-bang a acestora.

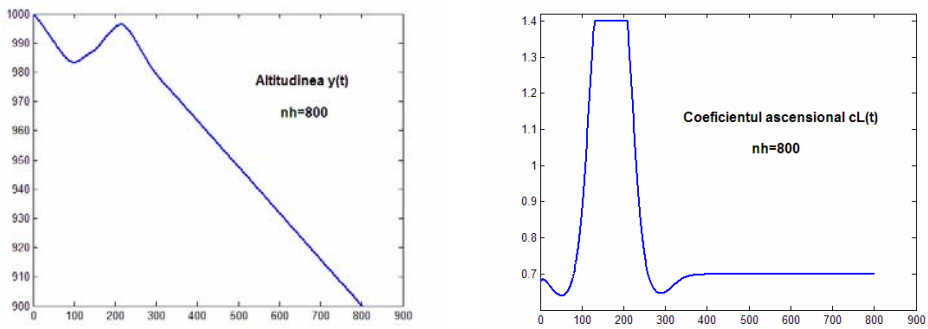

Fig. 8.2. Evoluția altitudinii $y(t)$ şi a coeficientului de ridicare aerodinamică $c_L(t)$.

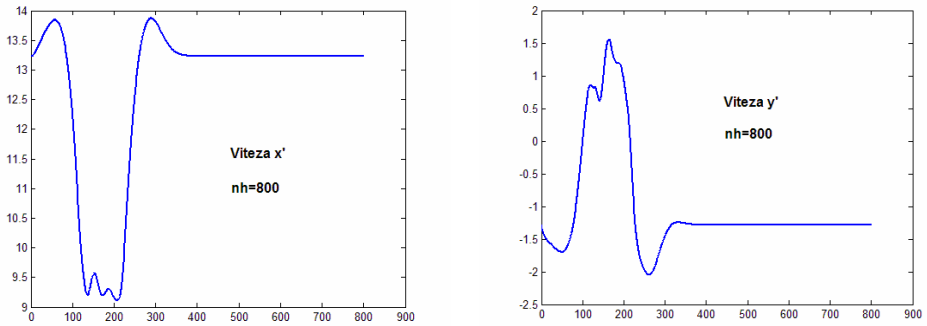

Fig. 8.3. Evoluția vitezelor $x'(t)$ şi $y'(t)$.

Aplicația G9 (*Torsion*)

Torsiunea elasto-plastică a unei bare. Problema constă în determinarea câmpului de eforturi într-o bară cilindrică infinit lungă.

Versiunea infinit dimensională a acestei probleme este următoarea.

$$\min\{q(v) : v \in K\},$$

unde $q : K \to R$ este funcția pătratică:

$$q(v) = \frac{1}{2}\int_D \|\nabla v(x)\|^2 \, \mathrm{d}x - c\int_D v(x)\mathrm{d}x$$

pentru o constantă oarecare c (unghiul de torsiune pe unitatea de lungime) şi D este un domeniu mărginit cu frontieră netedă (secțiunea transversală în bară). Mulțimea convexă K este definită ca:

$$K = \left\{ v \in H_0^1(D) : |v(x)| \le dist(x, \partial D), x \in D \right\},$$

unde $dist(.,\partial D)$ este distanța la frontiera lui D, şi $H_0^1(D)$ este spațiul Hilbert al tuturor funcțiilor cu suport compact în D astfel încât v şi $\|\nabla v\|^2$ aparțin lui $L^2(D)$. Această formulare, precum şi interpretarea fizică a acestei probleme este prezentată de Glowinski [1984, pp.41-55].

Aproximarea prin elemente finite a problemei se obține prin discretizarea lui D şi înlocuirea problemei de minimizare a lui q pe $H_0^1(D)$ prin minimizarea lui q pe mulțimea funcțiilor liniare pe porțiuni care satisfac restricțiile specificate de K, aşa cum este descris de Averick, Carter, Moré şi Xue [1994]. Se consideră $D=[0,1]\times[0,1]$ şi se utilizează o triangulație cu n_x, respectiv n_y puncte de discretizare în direcția coordonatelor. Pentru unghiul de torsiune pe unitatea de lungime se consideră $c=5$. Ca valori inițiale se consideră funcția de distanță $dist(x,\partial D)$ evaluată în punctele de discretizare ale domeniului.

Expresia GAMS a problemei este arătată în figura 9.1.

```
$ontext
Determine the stress potential in an infinitely long cylinder
when torsion is applied.
$offtext

$if not set nx $set nx 40
$if not set ny $set ny 40

sets nx grid points in 1st direction / x0*x%nx% /
     ny grid points in 2st direction / y0*y%ny% /

alias(nx,i),(ny,j);

parameters D(nx,ny) Distance to the boundary
           hx grid spacing for x
           hy grid spacing for y
           area area of triangle
           c some constant / 5.0 /;

hx := 1/(card(nx)-1);
hy := 1/(card(ny)-1);
area := 0.5*hx*hy;

D(i,j) := min(min(ord(i)-1,card(nx)-ord(i))*hx,
              min(ord(j)-1,card(ny)-ord(j))*hy);

variables v(nx,ny) the finite element approximation stress,
          linLower,
          linUpper,
          quadLower,
          quadUpper,
          stress;

Equations defLL,
          defLU,
          defQL,
          defQU,
          defstress;

defLL.. linLower =e= sum((nx(i+1),ny(j+1)), v[i+1,j] + v[i,j] +
                                            v[i,j+1]);
```

```
defLU.. linUpper =e= sum((nx(i-1),ny(j-1)), v[i,j] + v[i-1,j] +
                                            v[i,j-1]);

defQL.. quadLower =e= sum((nx(i+1),ny(j+1)),
                              sqr((v[i+1,j]-v[i,j])/hx) +
                              sqr((v[i,j+1]-v[i,j])/hy));

defQU.. quadUpper =e= sum((nx(i-1),ny(j-1)),
                              sqr((v[i,j]-v[i-1,j])/hx) +
                              sqr((v[i,j]-v[i,j-1])/hy));

defstress.. stress =e= area*( (quadLower + quadUpper)/2 -
                          c*(linLower + linUpper )/3);

model torsion / all /;

v.lo(i,j) = -d(i,j);
v.up(i,j) = d(i,j);
v.l (i,j) = d(i,j);

display d,hx,hy,area;

*$onecho >minos.opt
*  superbasics limit = 5000
*$offecho

*$onecho >bench.opt
* solvers conopt knitro minos.1
*$offecho

torsion.optfile=1;
torsion.workspace=120;
option nlp=conopt;
solve torsion minimizing stress using nlp;

file res1 /torsion.dat/
res1.pw=4000;
put res1;
loop(nx, loop(ny, put v.l(nx,ny):6:2; ); put /;); put /;

* End Torsion
```

Fig. 9.1. Expresia GAMS a aplicației G9 (Torsion).

Performanțele algoritmilor CONOPT, KNITRO şi MINOS pentru diferite valori ale lui n_x şi n_y sunt arătate în tabelele de mai jos.

Tabelul 9.1

$n_x = 40,\ n_y = 40,\ n = 1686$ (variabile), $m = 5$ (restricții)

	#iter	time	vfo
CONOPT	30	3.109	-0.4178327477
KNITRO	59	6.379	-0.4176095651
MINOS	1000	11.008	-0.4178327

Tabelul 9.2

$n_x = 51,\ n_y = 51,\ n = 2709$ (variabile), $m = 5$ (restricții)

	#iter	time	vfo
CONOPT	34	1.230	-0.4180876320
KNITRO	86	22.492	-0.4178122579

Tabelul 9.3

$n_x = 101,\ n_y = 101,\ n = 10409$ (variabile), $m = 5$ (restricții)

	#iter	time	vfo
CONOPT	60	8.621	-0.418391026
KNITRO	68	289.926	-0.4168094513

Tabelul 9.4

$n_x = 200,\ n_y = 200,\ n = 40406$ (variabile), $m = 5$ (restricții)

	#iter	time	vfo
CONOPT	109	85.191	-0.4184683991

În figura 9.2 se prezintă potențialul efortului în bara considerată pentru discretizarea $n_x = 40,\ n_y = 40$ şi $c = 5$.

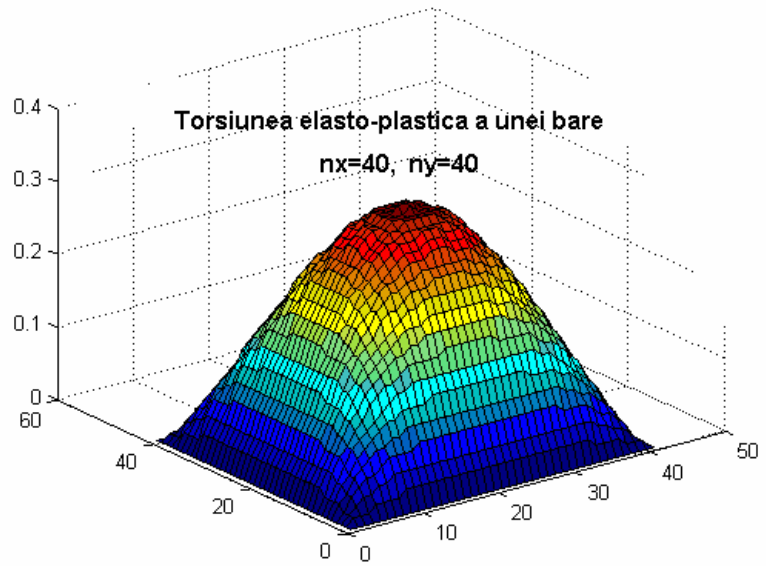

Fig.9.2. Soluția aplicației G9. $nx = 40,\ ny = 40$.

Astfel formulată problema ridică dificultăți majore algoritmilor de optimizare neliniară cu restricții. Chiar pentru dimensiuni modeste ale discretizării, algoritmul MINOS nu o poate rezolva. Şi algoritmul KNITRO pentru discretizări mai fine implică un timp de calcul total nerezonabil.

Problema se poate reformula ca una de optimizare fără restricţii (vezi [Andrei, 2009]). În acest caz, performanţele algoritmilor de optimizare fără restricţii sunt mult superioare. De exemplu algoritmul SCALCD [Andrei, 2007], [Andrei, 2009, pg.432], pentru discretizarea $n_x = 200$, $n_y = 200$, furnizează o soluţie optimă a acestei problemei în 33.64 secunde. Algoritmul L-BFGS [Andrei, 2009, pg. 537] pentru aceeaşi discretizare furnizează soluţia în 20.90 secunde, iar algoritmul TN (Newton trunchiat) [Andrei, 2009, pg. 571] în 19.75 secunde. Vedem imporatnţa formulării problemei şi a utilizării algoritmilor specializaţi.

Aplicaţia G10 (*Lagar*)
Distribuţia presiunii într-un lagăr de alunecare (cuzinet). Problema constă în determinarea distribuţiei presiunii într-un film subţire de lubrifiant între doi cilindri circulari.
Versiunea infinit-dimensională a problemei este:

$$min\{q(v) : v \in K\},$$

$$q(v) = \frac{1}{2}\int_D w_q(x)\|\nabla v(x)\|^2 \,\mathrm{d}x - \int_D w_l(x)v(x)\mathrm{d}x$$

cu

$$w_q(z_1, z_2) = (1 + \varepsilon\cos z_1)^3, \quad w_l(z_1, z_2) = \varepsilon\sin z_1,$$

pentru o anumită constantă $\varepsilon \in (0,1)$ şi $D = (0, 2\pi) \times (0, 2b)$ unde $b > 0$ este o constantă. Mulţimea convexă K este $K = \{v \in H_0^1(D) : v \in D, v \geq 0\}$. Aproximarea prin elemente finite a acestei probleme se obţine exact ca în aplicaţia de mai sus, unde de data aceasta $w_q(\xi_1, \xi_2) = (1 + \varepsilon\cos\xi_1)^3$ şi $w_l(\xi_1, \xi_2) = \varepsilon\sin\xi_1$.
Domeniul de admisibilitate este mulţimea

$$\Omega = \{v \in R^{n_x n_y} : v_{i,j} \geq 0\}.$$

Considerând $b = 10$ şi $\varepsilon = 0.1$, şi o discretizare $n_x \times n_y$ a domeniului $D = (0, 2\pi) \times (0, 2b)$, atunci pentru diferite valori a lui n_x şi n_y, performanţele algoritmilor CNOPT, KNITRO şi MINOS sunt prezentate în tabelele de mai jos. În figura 10.1 se prezintă expresia GAMS a acestei aplicaţii.

```
$ontext
Given the eccentricity e of the journal bearing, find the
pressure distribution in the lubricant separating the shaft
from the bearing.
$offtext

$if not set nx $set nx 40
$if not set ny $set ny 40

Set nx / 0*%nx% /
    ny / 0*%ny% /

alias (nx,i),(ny,j);
```

```
Scalar pi a famous constant
       b "grid is (0,2*pi)x(0,2*b)"  /10/
       e eccentricity                /0.1/
       hx, hy grid spacing
       area area of triangle;

pi = 2*arctan(inf);
hx = 2*pi/%nx%;
hy = 2*b/%ny%;
area = 0.5*hx*hy;

Parameter wq(nx);
wq(nx) = power(1+e*cos((ord(nx)-1)*hx),3);

Positive variable v(nx,ny);
Variable obj;

Equation defobj Objective function;

defobj.. obj =e= (hx*hy/12)*sum{(nx(i+1),ny(j+1)),
     (wq[i]+2*wq[i+1])*
     (sqr((v[i+1,j]-v[i,j])/hx) + sqr((v[i,j+1]-v[i,j])/hy))}+
     (hx*hy/12)*sum{(nx(i+1),ny(j+1)), (2*wq[i+1]+2*wq[i])*
     (sqr((v[i,j+1]-v[i+1,j+1])/hx) +
     sqr((v[i+1,j]-v[i+1,j+1])/hy))} -
     hx*hy*sum {(nx,ny), e*sin((ord(nx)-1)*hx)*v[nx,ny]};

* Starting point
v.l[nx,ny] = max(sin((ord(nx)-1)*hx),0);

v.fx[nx, '0'] = 0;
v.fx[nx,'%ny%'] = 0;
v.fx[ '0',ny] = 0;
v.fx['%nx%',ny] = 0;

model lagar /all/;

$onecho >minos.opt
  superbasic limit 5000
$offecho

$onecho >bench.opt
  solvers conopt knitro
$offecho

lagar.optfile=1;
lagar.workspace=125;
lagar.reslim=6000;
option nlp=bench;

solve lagar minimizing obj using nlp;
*file rez /lagar.dat/
*put rez;
*loop(i, loop(j, put v.l(i,j):6:2); put/;);put/;
* End Lagar
```

Fig. 10.1. Expresia GAMS a aplicației G10 (Lagar).

Tabelul 10.1

$n_x = 40,\ n_y = 40,\ n = 1682$ (variabile), $m = 1$ (restricții)

	#iter	time	vfo
CONOPT	29	6.121	-0.154791668
KNITRO	86	8.201	-0.154754911
MINOS	1300	13.590	-0.1547917

Tabelul 10.2

$n_x = 50,\ n_y = 50,\ n = 2602$ (variabile), $m = 1$ (restricții)

	#iter	time	vfo
CONOPT	56	22.133	-0.154820504
KNITRO	90	18.847	-0.154765045
MINOS	2100	50.379	-0.1548205

Tabelul 10.3

$n_x = 100,\ n_y = 100,\ n = 10202$ (variabile), $m = 1$ (restricții)

	#iter	time	vfo
CONOPT	38	6.012	-0.154839385
KNITRO	56	160.661	-0.153782131

Tabelul 10.4

$n_x = 200,\ n_y = 200,\ n = 40402$ (variabile), $m = 1$ (restricții)

	#iter	time	vfo
CONOPT	63	56.691	-0.154828784
KNITRO	90	3952.403	-0.154701894

Figura 10.2 prezintă distribuția presiunii în lagăr pentru discretizarea $n_x = 40$, $n_y = 40$ şi excentricitatea $\varepsilon = 0.1$.

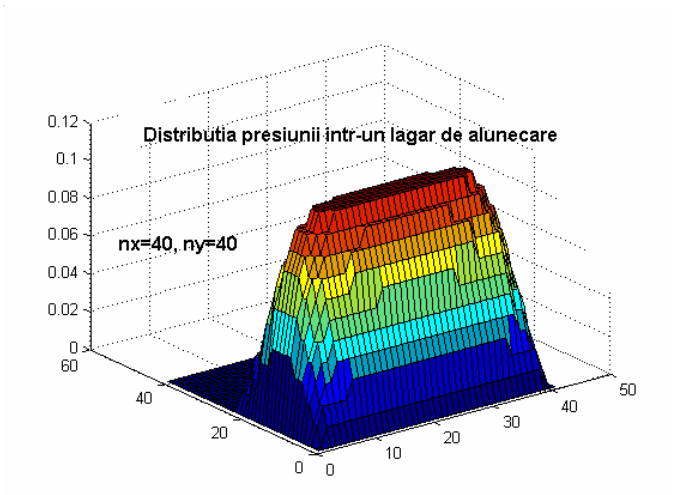

Fig. 10.2. Soluția aplicației G10 pentru $\varepsilon = 0.1$ şi $b = 10$.

Aplicația G11 (*Minsurf*)
Suprafața minimală cu obstacol. Să se determine o suprafață de arie minimă care zace deasupra unui obstacol cu condiții la frontieră date.

Problema constă în a determina suprafețele de arie minimă din $\mathbb{R}^3$, care au ca frontieră o curbă închisă dată. Aceasta a fost formulată de Lagrange în 1760, care pentru clasa suprafețelor de forma $z = z(x, y)$ a rerdus-o la rezolvarea ecuațiilor Euler-Lagrange pentru suprafețe minimale.

Presupunem că suprafața se poate reprezenta într-o formă neparametrică de tipul $v : \mathbb{R}^2 \to \mathbb{R}$, care satisface restricția $v \geq v_L$ pentru un anumit obstacol v_L dat. Ca atare, determinarea suprafeței cu arie minimă când condițiile pe frontiera unui domeniu convex D sunt cunoscute este o problemă infinit dimensională de forma:

$$min\left\{ f(v) : v \in K \right\},$$

unde $f : K \to R$ este funcționala

$$f(v) = \int_{\mathrm{D}} \left(1 + \left\|\nabla v(x)\right\|^2\right)^{1/2} \mathrm{d}x$$

şi mulțimea convexă K este definită prin:

$$K = \left\{ v \in H^1(D) : v(x) = v_D(x),\, x \in \partial D,\, v(x) \geq v_L(x),\, x \in D \right\},$$

unde $H^1(D)$ este spațiul funcțiilor derivabile în $L^2(D)$, funcția $v_D : \partial D \to \mathbb{R}$ defineşte datele de la frontieră şi $v_L : D \to \mathbb{R}$ este obstacolul. Presupunem că pe frontiera ∂D a domeniului D $v_L \leq v_D$.

În această aplicație presupunem că

$$v_D = \begin{cases} 1 - (2x-1)^2, & y = 0,1 \\ 0, & (0,1), \end{cases} \qquad v_L = \begin{cases} 1, & |x - 0.5| \leq 0.25, |y - 0.5| \leq 0.25 \\ 0, & \text{altfel.} \end{cases}$$

O aproximație prin elemente finite a acestei probleme se obține prin minimizarea lui f pe spațiul funcțiilor liniare pe porțiuni v cu valorile $v_{i,j}$ în nodurile unei triangulații a lui D. Considerăm $D = [0,1] \times [0,1]$ cu n_x şi n_y puncte de discretizare în direcția axelor de coordonate. Ca soluție inițială considerăm funcția $1 - (2x-1)^2$ evaluată în nodurile rețelei de discretizare.

Expresia GAMS a acestei aplicații este prezentată în figura 11.1.

```
$ontext
Find the surface with minimal area, given boundary conditions,
and above an obstacle.
$offtext

$if not set nx $set nx 50
$if not set ny $set ny 50

sets nx grid points in 1st direction / x0*x%nx% /
     ny grid points in 2st direction / y0*y%ny% /
```

```
alias(nx,i),(ny,j);

parameters hx grid spacing for x
           hy grid spacing for y
           area area of triangle;

hx := 1/(card(nx)-1);
hy := 1/(card(ny)-1);
area := 0.5*hx*hy;

variables v(nx,ny) defines the finite element approximation
          surf;
positive variable v;

equation defsurf;
defsurf..
surf/area =e= sum((nx(i+1),ny(j+1)),
         sqrt(1+sqr((v[i+1,j]-v[i,j])/hx)+sqr((v[i,j+1]-
v[i,j])/hy))) +
               sum((nx(i-1),ny(j-1)),
         sqrt(1+sqr((v[i-1,j]-v[i,j])/hx)+sqr((v[i,j-1]-
v[i,j])/hy)));

v.fx['x0' ,j] = 0;
v.fx['x%nx%',j] = 0;
v.fx[i,'y0' ] = 1 - sqr(2*(ord(i)-1)*hx-1);
v.fx[i,'y%ny%'] = 1 - sqr(2*(ord(i)-1)*hx-1);

v.lo(i,j)$(((ord(i)-1) >= floor(0.25/hx) and (ord(i)-1)
                       <= ceil(0.75/hx)) and ((ord(j)-1)
                       >= floor(0.25/hy) and (ord(j)-1)
                       <= ceil(0.75/hy))) = 1;
v.l(i,j) = 1 - sqr(2*(ord(i)-1)*hx-1);

model minsurf / all /;

$onecho >minos.opt
  superbasic limit 5000
$offecho

$onecho >bench.opt
  solvers conopt knitro minos.1
$offecho

minsurf.optfile=1;
minsurf.workspace=125;
minsurf.reslim=6000;
option nlp=bench;
solve minsurf minimizing surf using nlp;

*file rez /minsurf.dat/
*put rez;
*loop(i, loop(j, put v.l(i,j):6:2); put/;);put/;

* End Minsurf
```

Fig. 11.1. Expresia GAMS a aplicației G11 (Minsurf).

Performanțele algoritmilor CONOPT, KNITRO şi MINOS pentru diferite valori ale lui n_x şi n_y sunt date în tabelele următoare.

Tabelul 11.1

$n_x = 40,\ n_y = 40,\ n = 1682$ (variabile), $m = 1$ (restricții)

	#iter	**time**	**vfo**
CONOPT	14	5.148	2.487506264
KNITRO	640	75.919	2.487573608
MINOS	1300	19.055	2.487506

Tabelul 11.2

$n_x = 50,\ n_y = 50,\ n = 2602$ (variabile), $m = 1$ (restricții)

	#iter	**time**	**vfo**
CONOPT	23	15.434	2.505616501
KNITRO	995	250.920	2.505724225
MINOS	2100	63.148	2.505617

Tabelul 11.3

$n_x = 60,\ n_y = 60,\ n = 3722$ (variabile), $m = 1$ (restricții)

	#iter	**time**	**vfo**
CONOPT	27	1.180	2.485698315
KNITRO	604	296.506	2.48583837
MINOS	3262	197.906	2.485698

În figura 11.2 se prezintă soluția acestei aplicații.

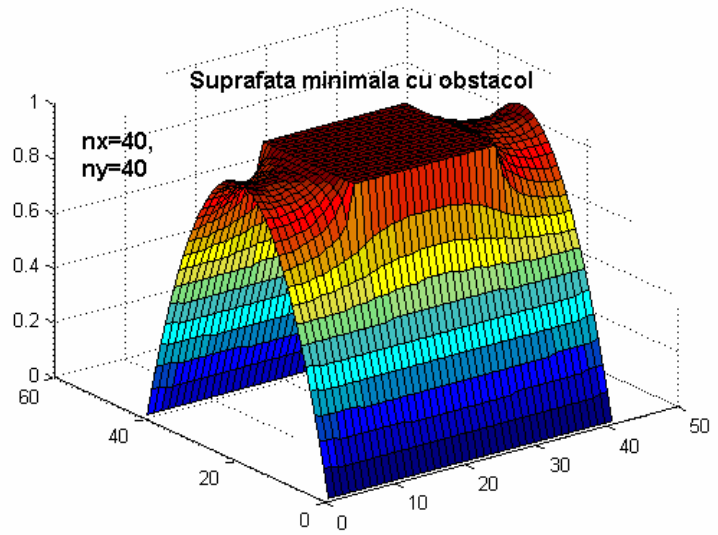

Fig. 11.2. Soluția aplicației G11, $n_x = 40,\ n_y = 40$.

Aplicația G12 (*Circle*)
Să se determine cercul de rază minimă care conține un număr dat de puncte de coordonate (x_i, y_i).

Fie cercul de rază r şi centru de coordonate (a,b), care conține cele n_p puncte date de coordonate (x_i, y_i). Atunci, modelul matematic al acestei probleme este:

$$\min_{a,b,r} r$$

referitor la:

$$(x_i - a)^2 + (y_i - b)^2 \le r^2, \quad i = 1,\ldots,n_p,$$
$$r \ge 0.$$

Punctul inițial are o importanță crucială în eficientizarea funcționării optimizatoarelor. De aceea ca punct inițial considerăm: $x_{\min} = \min_{1\le i\le n_p} x_i$, $x_{\max} = \max_{1\le i\le n_p} x_i$, $y_{\min} = \min_{1\le i\le n_p} y_i$ şi $y_{\max} = \max_{1\le i\le n_p} y_i$, cu care se calculează o estimație a punctului inițial sub forma:

$$a = (x_{\min} + x_{\max})/2, \qquad b = (y_{\min} + y_{\max})/2,$$
$$r = \sqrt{(a - x_{\min})^2 + (b - y_{\min})^2}.$$

Expresia GAMS a acestei aplicații este arătată în figura 12.1.

```
$ontext
The circle problem.
Find the smallest circle that contain a number of points.
$offtext

set      i       points /p1*p100/;

parameters
         x(i)   'x coordinates'
         y(i)   'y coordinates' ;

* Consider random data for these coordinates
         x(i) = uniform(1,10);
         y(i) = uniform(1,10);

Variables
         a  'x coordinate of center of circle'
         b  'y coordinate of center of circle'
         r  'radius' ;

equations
         e(i)  'points must be inside circle' ;

e(i).. sqr(x(i)-a) + sqr(y(i)-b) =l= sqr(r);

*  Compute an initial point
scalars xmin, ymin, xmax, ymax;
   xmin = smin(i,x(i));
   ymin = smin(i,y(i));
```

```
   xmax = smax(i,x(i));
   ymax = smax(i,y(i));
   a.l = (xmin+xmax)/2;
   b.l = (ymin+ymax)/2;
   r.l = sqrt( sqr(a.l-xmin) + sqr(b.l-ymin) );

model circle /all/ ;

$onecho >bench.opt
  solvers conopt knitro minos
$offecho

circle.optfile=1;
option nlp=bench;

solve circle using nlp minimizing r;
* End circle
```

Fig. 12.1. Expresia GAMS a aplicației G12 (Circle).

Performanțele algoritmilor CNOPT, KNITRO şi MINOS sunt prezentate în tabelele de mai jos.

Tabelul 12.1

$n_p = 100,\ \ n = 3$ (variabile), $m = 100$ (restricții)

	#iter	time	vfo
CONOPT	6	0.023	5.42768655
KNITRO	52	0.280	5.42768752
MINOS	8	0.078	5.427687

Tabelul 12.2

$n_p = 500,\ \ n = 3$ (variabile), $m = 500$ (restricții)

	#iter	time	vfo
CONOPT	7	0.023	6.01176788
KNITRO	49	0.250	6.01176889
MINOS	7	0.133	6.011768

Tabelul 12.3

$n_p = 1000,\ \ n = 3$ (variabile), $m = 1000$ (restricții)

	#iter	time	vfo
CONOPT	7	0.117	6.21321892
KNITRO	52	0.791	6.21321955
MINOS	6	0.055	6.213219

Problema este interesantă deoarece constituie un bun exemplu privind importanța alegerii punctului inițial cu care se demarează optimizarea. Dacă punctul inițial este diferit de cel pe care l-am selectat mai sus, atunci toți algoritmii considerați în acest studiu au dificultăți în a determina soluția problemei. De exemplu, pentru $n_p = 100$, KNITRO nu poate determina soluția problemei.

Aplicația G13 (*Rezervor*)
Optimizarea Funcționării a Două Rezervoare.
Să considerăm situația în care dispunem de două rezervoare de volum S_1 şi respectiv S_2 . În primul rezervor intră apă conform unui hidrograf dat q şi iese apă în baza unei cereri r precizate de–a lungul unui interval de timp. Al doilea rezervor primeşte apă în cantitatea q_2 din primul printr-o valvă care se deschide când volumul de apă din primul rezervor depăşeşte volumul celui de–al doilea. În acelaşi timp, acest al doilea rezervor eliberează apă în primul, în cantitatea r_2 , de îndată ce volumul de apă din primul rezervor nu reuşeşte să satisfacă cererea de apă de la un moment dat, figura 13.1 [Andrei, 2004d]. Problema este de a *optimiza funcționarea acestei configurații de rezervoare astfel încât să se minimizeze cantitatea de apă eliberată din rezervorul* S_2 .

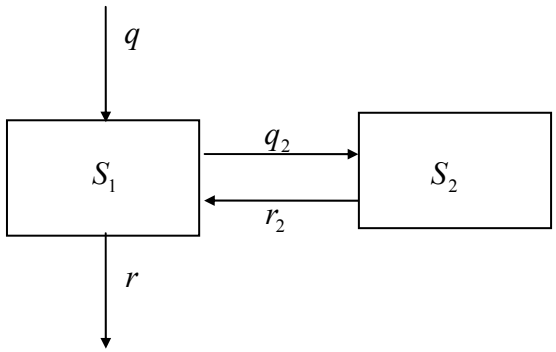

Fig. 13.1. Sistemul de rezervoare şi fluxurile de apă.

Modelul matematic care descrie funcționarea celor două rezervoare este următorul:

$$min \sum_{2}^{N} r_2(t)$$

referitor la:

$$S_1(t) = S_1(t-1) + q(t) + r_2(t) - q_2(t) - r(t),$$
$$S_2(t) = S_2(t-1) + q_2(t) - r_2(t),$$
$$S_1(t) = S_2(t) \text{ dacă } q(t) > 0 \text{ şi } r_2(t) = 0,$$
$$S_1^{\min} \le S_1 \le S_1^{\max},$$
$$S_2^{\min} \le S_2 \le S_2^{\max},$$
$$0 \le q_2(t) \le q_{\max},$$
$$0 \le r_2(t) \le q_{\max},$$
$$t = 1, \ldots, N,$$

unde

$$S_1(1) = S_1^0, \quad S_2(1) = S_2^0,$$
$$q(t) = f(t),$$
$$r(t) = g(t), \quad t = 1, \ldots, N+1,$$

în care S_1^0 şi S_2^0 sunt cantităţile iniţiale de apă din cele două rezervoare, iar $f(t)$ şi $g(t)$ sunt cantităţile (debitele) de apă care intră şi respectiv care părăseşte primul rezervor, cunoscute pe întreaga perioadă de timp.

Considerând o evoluţie anuală $(N = 12)$, în care variabilele sunt monitorizate lunar, precum şi o serie de date concrete, expresia GAMS a acestui model este prezentată în figura 13.2.

```
$Ontext
Optimizarea funcționarii a două rezervoare cuplate. Al doilea rezervor
este considerat de ajutor. Când din primul rezervor există o cerere de
apă prea mare, atunci intervine cel de-al doilea rezervor din care se
ia deficitul de apă.
$Offtext

sets n rezervoare /rez1, rez2/;

sets t time /ian, feb, mar, apr, mai, jun, jul,
             aug, sep, oct, nov, dec, enda /

tt(t) /ian/;

table q(n,t) debitul de apa care intra in rezervorul rez1 (mil.m3)
      ian  feb  mar  apr  mai  jun  jul  aug  sep  oct  nov  dec  enda
rez1  128  125  234  360  541  645  807  512  267  210  981  928  250;

table r(n,t) debitul de apa cerut din rezervorul rez1 (mil.m3)
      ian  feb  mar  apr  mai  jun  jul  aug  sep  oct  nov  dec  enda
rez1  100  150  200  500  222  700  333  333  300  250  250  250  200;

variables q2(t),
          r2(t),
          s(n,t),
          obj;

equation
bal1(n,t)   balanta in rezervorul S1
bal2(n,t)   balanta in rezervorul S2
dec(n,t)    decizii de umplere a rezervoarelor
objf        functia obiectiv;

bal1(n,t)$(not tt(t))..
   s('rez1',t)-s('rez1',t-1) =E= Q('rez1',t)+r2(t)-
       q2(t)-r('rez1',t);

bal2(n,t)$(not tt(t))..
   s('rez2',t)-s('rez2',t-1) =E= q2(t)-r2(t);

dec(n,t)$(not tt(t))..
   (s('rez2',t)-s('rez1',t)) -
   (s('rez2',t)-s('rez1',t)) * (1.0-q2(t)/(q2(t)+0.000001)) =E=
       0.0;

objf.. obj =E= sum(t$(not tt(t)),r2(t));

* Limite asupra variabilelor, condițiile inițiale
s.lo('rez1',t)=1150;
s.up('rez1',t)=4590;
s.fx('rez1','ian')=1200;

s.lo('rez2',t)=100;
```

```
s.up('rez2',t)=4590;
s.fx('rez2','ian')=1200;

r2.up(t)=1500;
r2.lo(t)=0.0;

q2.up(t)=1500;
q2.lo(t)=0.0;
q2.l(t)=0.00001;

option optcr =0.000001;

model rezervor /all/;

solve rezervor using nlp minimizing obj;

parameter a(t);
  a(t)=(1-(q2.l(t)/(abs(q2.l(t))+0.00000001)));

* Afişarea rezultatelor optimizării
file res /rezerv.txt/
put res;
put "obiectiv = ", obj.l:10:5; put /;
put /"======================================================"/;
put /"            a      q-rezl     r-rezl    q2(t)   r2(t)    s-rezl    ds-
rezl    s-rezl2   ds-rezl2     "/;

loop (t $(ord(t) ne card(t)),
  put t.tl:7, a(t):5:2, q('rezl',t):10:2, r('rezl',t):10:2,
      q2.l(t):7:1, r2.l(t):7:1,
      s.l('rezl',t):10:2, (s.l('rezl',t)-s.l('rezl',t-1)):10:2,
      s.l('rezl2',t):10:2, (s.l('rezl2',t)-s.l('rezl2',t-1)):10:2 /; );
put /"======================================================"/;
* End of rezervor
```

Fig. 13.2. Expresia GAMS a aplicației G13 (Rezervor).

Soluția problemei de optimizare care se referă la minimizarea cantității de apă eliberată de rezervorul S_2, obținută cu MINOS, este prezentată în tabelul de mai jos. Observăm că $r_2(t)$ este zero cu excepția lunii februarie când are o valoare nenulă. Valoarea minimă a funcției obiectiv a problemei este egală cu 81.00, adică exact valoarea lui $r_2(feb)$.

Tabelul 13.1.

Soluția modelului

obiectiv = 81.00000

	a	q-rezl	r-rezl	q2(t)	r2(t)	s-rezl	ds-rezl	s-rezl2	ds-rezl2
ian	0.00	128.00	100.00	0.0	0.0	1200.00	1200.00	1200.00	1200.00
feb	0.00	125.00	150.00	0.0	81.0	1256.00	56.00	1119.00	-81.00
mar	0.00	234.00	200.00	0.0	0.0	1290.00	34.00	1119.00	0.00
apr	0.00	360.00	500.00	0.0	0.0	1150.00	-140.00	1119.00	0.00
mai	0.00	541.00	222.00	175.0	0.0	1294.00	144.00	1294.00	175.00
jun	0.00	645.00	700.00	0.0	0.0	1239.00	-55.00	1294.00	0.00
jul	0.00	807.00	333.00	209.5	0.0	1503.50	264.50	1503.50	209.50
aug	0.00	512.00	333.00	89.5	0.0	1593.00	89.50	1593.00	89.50
sep	0.00	267.00	300.00	0.0	0.0	1560.00	-33.00	1593.00	0.00
oct	0.00	210.00	250.00	0.0	0.0	1520.00	-40.00	1593.00	0.00
nov	0.00	981.00	250.00	0.0	0.0	2251.00	731.00	1593.00	0.00
dec	0.00	928.00	250.00	0.0	0.0	2929.00	678.00	1593.00	0.00

Problema are 51 de variabile şi 73 de restricții. CONOPT necesită 6 iterații, iar MINOS doar 2 iterații. KNITRO nu poate rezolva problema.
În figura 13.3 se arată evoluția stocului de apă din cele două rezervoare, precum şi debitele de apă $q_2(t)$ şi $r_2(t)$ dintre cele două rezervoare.

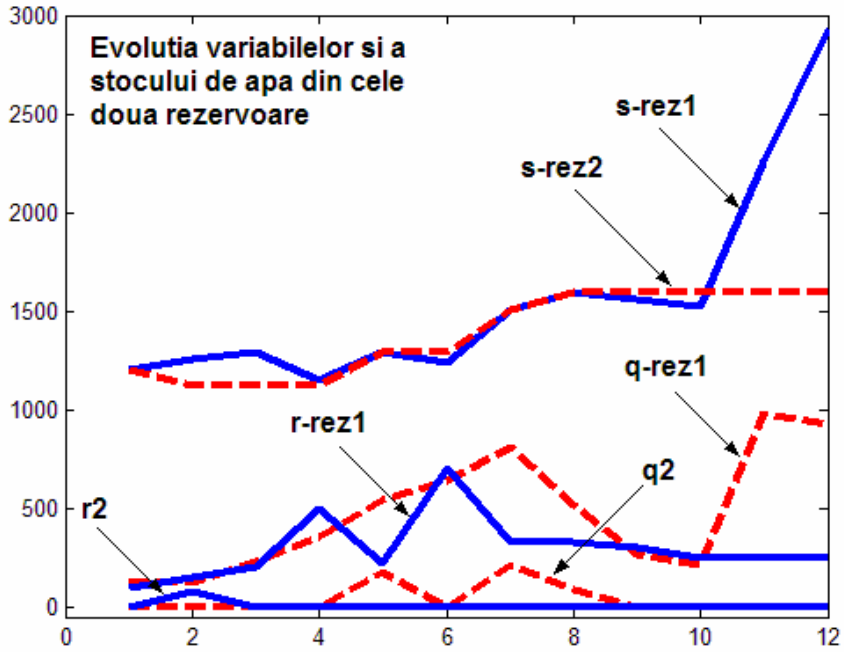

Fig. 13.3. Evoluția variabilelor şi a stocului de apă din cele două rezervoare.

Aplicația G14 (*Lacuri*)
Optimizarea funcționării bazinelor hidrografice ale unor râuri care alimentează câteva lacuri şi rezervoare (bazine).
Un bazin hidrografic include mai multe râuri, lacuri, rezervoare (bazine) construite artificial, precum şi utilizatori ai apei care în principal o utilizează la irigații. În situații normale problema managementului unui bazin hidrografic se referă la satisfacerea nevoilor de apă a unor utilizatori, de-a lungul unei perioade de timp (de obicei un an).
Pentru a modela un bazin hidrografic acesta se poate reprezenta ca un graf. Nodurile acestui graf sunt anumite elemente ale bazinului, iar laturile sunt canalele de comunicație între noduri. Fiecare nod este caracterizat de o regulă de modificare a debitului de apă care constă din balanța de substanță (apă). Să considerăm deci următorul exemplu de bazin hidrografic [Andrei, 2005], prezentat în figura 14.1.
Evident că se pot prezenta situații mai complexe, cu mai multe elemente şi cu mai multe conexiuni între acestea. Totuşi, cazul prezentat este suficient de general pentru a ilustra tehnica de modelare şi reprezentarea acestuia în limbajul GAMS.

Pentru reprezentarea matematică a unui bazin hidrografic considerăm următoarele mulțimi de noduri:

nn - noduri ale rezervoarelor (Rez1, Rez2,...,Rez5),
ns - noduri ale surselor de apă, ale râurilor (Sursa1, Sursa2),
nr - noduri ale utilizatorilor (Utilizator1, Utilizator2, Evacuare),
nrr - noduri ale utilizatorilor (Utilizator1, Utilizator2)
nl - noduri ale lacurilor (Lac1, Lac2).

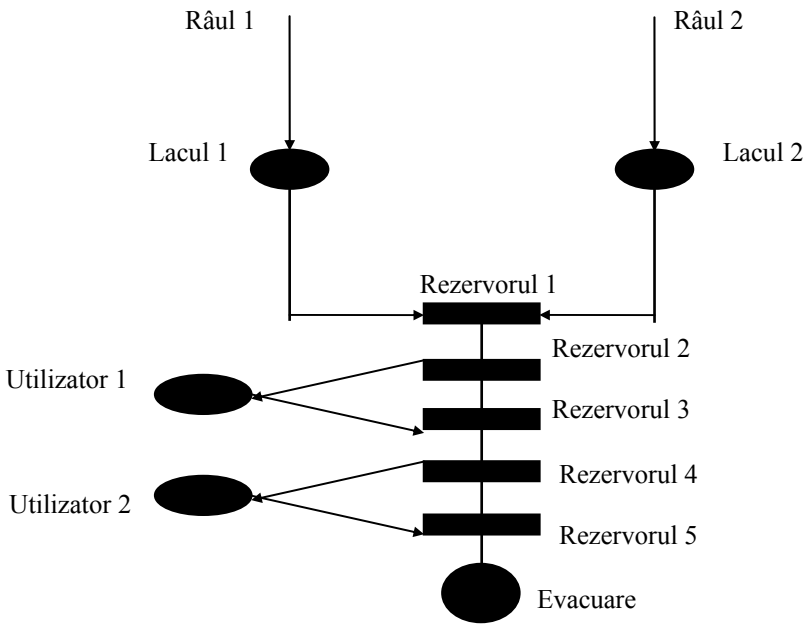

Fig. 14.1. Bazin hidrografic cu 2 râuri, 2 lacuri, 5 rezervoare şi 2 utilizatori.

Considerăm că orizontul de optimizare este de un an şi că dispunem de date privind debitul de apă din râuri (*sursa*) în lacuri, ca valori medii lunare, ca în tabelul de mai jos:

	Ian	**Feb**	**Mar**	**Apr**	**Mai**	**Iun**
Râu1	98	115	244	390	641	754
Râu2	29	49	78	121	198	144
	Iul	**Aug**	**Sep**	**Oct**	**Nov**	**Dec**
Râu1	807	512	367	210	181	128
Râu2	105	98	79	72	45	29

Evident că aceste valori medii sunt cunoscute în urma unor măsurători efectuate de-a lungul anilor şi care presupune o situație normală a regimului hidrografic.

Definim:

$R(n,t)$ - ieşirea din nodul n la momentul de timp t (luna t) (m^3),

$Q(n,t)$ - intrarea în nodul n la momentul de timp t (luna t) (m^3),

$S(n,t)$ - stocul (volumul) de apă din nodul n la momentul de timp t (luna t) (m^3),

$U(n,t)$ - Cantitatea de apă pentru utilizatorul n la momentul de timp t (luna t) (m^3),

Cu acestea ecuația lacurilor este:

$$S(n,t) = S(n,t-1) + \sum_{n \in in} Q(n,t) - \sum_{n \in out} R(n,t), \quad n \in nl,$$

unde in reprezintă numărul de intrări în lac, iar out numărul de ieşiri din lac. Se presupune că se cunoaşte stocul inițial de apă din lacuri.

Ecuația rezervoarelor este următoarea:

$$\sum_{n \in out} R(n,t) = \sum_{n \in in} Q(n,t), \quad n \in nn,$$

Nodurile de irigare (utilizator) sunt caracterizate de ecuația:

$$R(n,t) = ret_n \sum_{n \in in} Q(n,t), \quad n \in nrr,$$

unde ret_n este un coeficient de utilizare asociat fiecărui nod de irigare.

Funcția obiectiv se exprimă în funcție de nevoile de apă la nodurile utilizator. Presupunem că de-a lungul celor 12 luni cele două noduri utilizator au nevoie de următoarele debite (*cerere*) de apă:

	Ian	**Feb**	**Mar**	**Apr**	**Mai**	**Iun**
U1	0	0	10	64.5	189.8	184.4
U2	0	0	10	13.5	15	22.1
	Iul	**Aug**	**Sep**	**Oct**	**Nov**	**Dec**
U1	243.7	200.9	99.5	0	0	0
U2	26	24.9	13	0	0	0

Atunci, funcția obiectiv se scrie sub forma:

$$min \sum_n \sum_t \left(U(n,t) - cerere(n,t) \right)^2.$$

Expresia GAMS a modelului bazinului hidrografic din figura 14.1 este arătată in figura 14.2.

```
$ontext
Octombrie 6, 2005
Optimizarea functionarii bazinelor hidrografice ale unor rauri care
contine cateva lacuri. In acest exemplu am considerat doua surse
(izvoare, rauri) care alimenteaza doua lacuri. Din aceste doua lacuri
apa se duce intr-un lant de 5 rezervoare puse in cascada. Din
rezervorul 2 se ia apa pentru Utilizatorul 1, iar din rezervorul 4 se
ia apa pentru Utilizatorul 2. Utilizatorii pot folosi apa pentru
```

```
irigatii, de exemplu. Din ultimul rezervor (al 5-lea) apa este
deversata intr-un evacuator, care fizic poate fi chiar un parau care
apartine bazinului hidrografic considerat. Pentru a avea control asupra
bazinului, evacuatorul il vom considera ca un Utilizator (special).
Debitul de apa (m3 per secunda) care vine din surse (izvoare, rauri)
este cunoscut pe fiecare luna, ca valori medii.
$offtext

SET n noduri
  / Sursa_1, Sursa_2,
    Rez_1*Rez_5,
    U_1, U_2,
    Lac_1, Lac_2,
    Evacuare /;

ALIAS (n,n1);

SET
nn(n)    Rezervoare      / Rez_1*Rez_5/
ns(n)    Surse           / Sursa_1*Sursa_2 /
nr(n)    U               / U_1, U_2, Evacuare /
nrr(n)   Irigari         / U_1, U_2 /
nl(n)    Lacuri          / Lac_1, Lac_2 /;

SET n_from_n(n,n1) Topologia retelei de lacuri - rezervoare - evacuare.
*                  De unde se ia apa. De exemplu: Lacul 1 ia apa din
*                  sursa 1, Lacul 2 din sursa 2, etc.
/ Lac_1.Sursa_1,
  Lac_2.Sursa_2,
  Rez_1.Lac_1,
  Rez_1.Lac_2,
  Rez_2.Rez_1,
  U_1.Rez_2,
  Rez_3.Rez_2,
  Rez_3.U_1,
  Rez_4.Rez_3,
  U_2.Rez_4,
  Rez_5.Rez_4,
  Rez_5.U_2,
  Evacuare.Rez_5 /;

SET n_to_nr(n,n1) Topologia retelei care alimenteaza Uii.
/ Rez_2.U_1,
  Rez_4.U_2,
  Rez_5.Evacuare /;

SET t luni /Ian,Feb,Mar,Apr,Mai,Iun,Iul,Aug,Sep,Oct,Nov,Dec /;

PARAMETER Ini_S(n) Cantitatea initiala de apa din Lacuri. (m3)
/ Lac_1  1000,
  Lac_2   300 /;

PARAMETER ret(n) Coeficientii de curgere
/ U_1           0.5,
  U_2           0.5,
  Evacuare      0.0 /;

TABLE Sursa(n,t) Debitul de apa (m3 per sec) din surse (izvoare -
rauri)
         Ian  Feb  Mar  Apr  Mai  Iun  Iul  Aug  Sep  Oct  Nov  Dec
Sursa_1   98  115  244  390  641  754  807  512  367  210  181  128
Sursa_2   29   49   78  121  198  144  105   98   79   72   45   29
```

```
TABLE Cerere(n,t) Cererea (debitul) de apa (m3 per sec) ceruta de Ui.
         Ian   Feb   Mar     Apr    Mai    Iun    Iul    Aug   Sep   Oct   Nov
Dec
U_1         0     0    10    64.5 189.8 184.4 243.7 200.9 99.5     0     0
0
U_2         0     0    10    13.5  15.0  22.1  26.0  24.9 13.0     0     0
0
Evacuare 500   500   500     100    100    100    100    500   500   500   500
500

POSITIVE VARIABLES
U(n,t)       Cantitatea de apa de la nodul n in perioada t (m3)
q(n,t)       Intrare in nodul n in perioada t (m3)
r(n,t)       Iesire din nodul n in perioada t (m3)
s(n,t)       Stoc (volumul) de apa din nodul n in perioada t (m3);

Variable obj;

* Marginile superioare ale Uilor.
     U.up(n,t) = Cerere(n,t);
* Marginile superioare ale Lacurilor.
     s.up('Lac_1',t) = 1000;
     s.up('Lac_2',t) =  300;
* Stocul final in Lacuri (din decembrie).
     s.lo('Lac_1','Dec') = 1000;
     s.lo('Lac_2','Dec') =  300;

EQUATIONS
R_no(n,t)        Noduri simple
R_ns(n,t)        Surse (izvoare)
R_nr(n,t)        Ui (de irigare)
R_nl(n,t)        Rezervoare
R_nn(n,t)        Noduri simple
Objective        Functia obiectiv;

* Ecuatiile pentru nodurile simple: iesire = intrare
R_no(n,t)$(nn(n)).. R(n,t) =e= q(n,t);

* Ecuatiile pentru surse: iesire = Sursa
R_ns(n,t)$(ns(n)).. R(n,t) =e= Sursa(n,t);

* Ecuatiile Uilor (de irigare) iesire = coef * U
R_nr(n,t)$(nr(n)).. R(n,t) =e= ret(n) * U(n,t);

* Ecuatiile Rezervoarelor: iesire = bilantul de masa (apa)
R_nl(n,t)$(nl(n)).. S(n,t) =e= Ini_S(n)$(ord(t) eq 1)
                         + s(n,t-1)$(ord(t) gt 1) + Q(n,t) - R(n,t);

* Ecuatiile nodurilor simple: intrare = suma iesirilor din rezervoarele
* din amonte minus suma iesirilor din rezervoare
R_nn(n,t).. Q(n,t) =e= sum(n1$(n_from_n(n,n1)),  R(n1,t))
                      - sum(n1$(n_to_nr(n,n1)) ,U(n1,t));

*Functia obiectiv
objective.. obj =e= sum(t,
                    sum(n$nr(n), power((U(n,t) - Cerere(n,t)),2) ));

MODEL Lacuri /all/;
SOLVE Lacuri USING NLP minimizing obj;

* Afisarea rezultatelor optimizarii modelului de lacuri
file Rezultate /lac1.txt /
put Rezultate;
```

```
put "Nod        Utiliz_1    Cerere_1    Utiliz_2    Cerere_2    Evacuare
Cerere"/;
loop((t), put t.tl:6,
   U.l('U_1',t):11.2, Cerere('U_1',t):11.2,
   U.l('U_2',t):11.2, Cerere('U_2',t):11.2,
   U.l('Evacuare',t):  11.2, Cerere('Evacuare',t)  :11.2 /;);

put /"---------------------------------- Lacuri-----------------------
-------- "/;
put "                          Lac 1                              Lac 2
"/;
put "             intrare        stoc      iesire    intrare        stoc
iesire "/;
put "                 Q-1         S-1         R-1         Q-2         S-2
R-2 "/;
loop((t), put t.tl:8,
   Q.L('Lac_1',t):11.2, S.L('Lac_1',t):11.2, R.L('Lac_1',t):11.2,
   Q.L('Lac_2',t):11.2, S.L('Lac_2',t):11.2, R.L('Lac_2',t):11.2 /;);
* End Lacuri
```

Fig. 14.2. Expresia GAMS a aplicației G14 (Lacuri).

Rezultatele optimizării sunt prezentate în tabelele de mai jos.

Tabelul 14.1.
Soluția modelului.

Nod	Utiliz_1	Cerere_1	Utiliz_2	Cerere_2	Evacuare	Cerere
Ian	0.00	0.00	0.00	0.00	500.00	500.00
Feb	0.00	0.00	0.00	0.00	500.00	500.00
Mar	10.00	10.00	10.00	10.00	500.00	500.00
Apr	64.50	64.50	13.50	13.50	100.00	100.00
Mai	189.80	189.80	15.00	15.00	100.00	100.00
Iun	184.40	184.40	22.10	22.10	100.00	100.00
Iul	243.70	243.70	26.00	26.00	100.00	100.00
Aug	199.93	200.90	23.93	24.90	498.07	500.00
Sep	58.00	99.50	0.00	13.00	417.00	500.00
Oct	0.00	0.00	0.00	0.00	282.00	500.00
Nov	0.00	0.00	0.00	0.00	226.00	500.00
Dec	0.00	0.00	0.00	0.00	157.00	500.00

Tabelul 14.2.
Evoluția variabilelor asocoate lacurilor

	Lacuri					
	Lac 1			Lac 2		
	intrare	stoc	iesire	intrare	stoc	iesire
	Q-1	S-1	R-1	Q-2	S-2	R-2
Ian	98.00	524.00	574.00	29.00	0.00	329.00
Feb	115.00	188.00	451.00	49.00	0.00	49.00
Mar	244.00	0.00	432.00	78.00	0.00	78.00
Apr	390.00	0.00	390.00	121.00	0.00	121.00
Mai	641.00	0.00	641.00	198.00	51.00	147.00
Iun	754.00	436.70	317.30	144.00	195.00	0.00
Iul	807.00	1000.00	243.70	105.00	300.00	0.00
Aug	512.00	1000.00	512.00	98.00	300.00	98.00
Sep	367.00	1000.00	367.00	79.00	300.00	79.00
Oct	210.00	1000.00	210.00	72.00	300.00	72.00
Nov	181.00	1000.00	181.00	45.00	300.00	45.00
Dec	128.00	1000.00	128.00	29.00	300.00	29.00

Figurile 14.3 şi 14.4 ilustrează evoluția variabilelor (intrări, stocuri, ieşiri) asociate celor două lacuri.

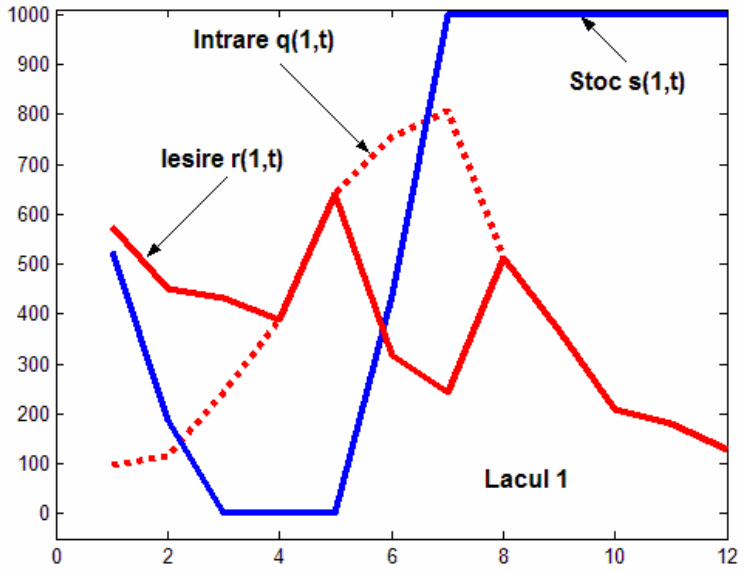

Fig. 14.3. Evluția variabilelor asociate lacului 1.

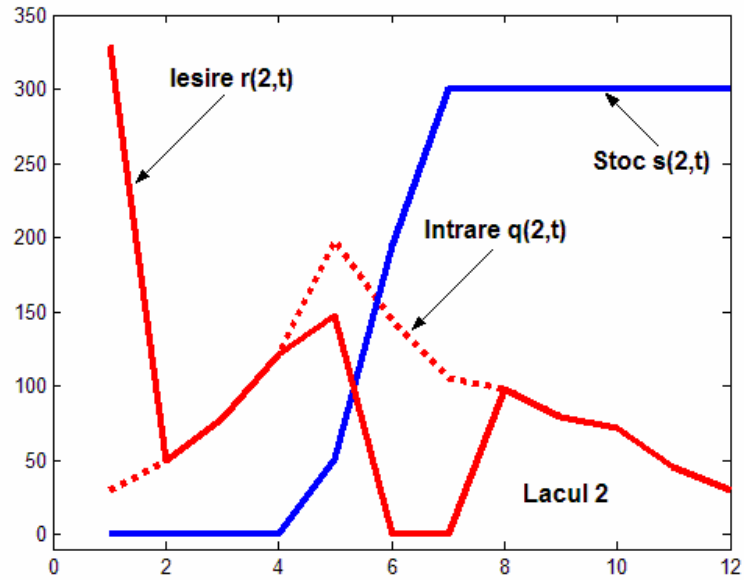

Fig. 14.4. Evluția variabilelor asociate lacului 2.

Performanțele algoritmilor CONOPT, KNITRO şi MINOS sunt arătate în tabelul 14.3.

Tabelul 14.3
$n = 349$ (variabile), $m = 289$ (restricții)

	#iter	time	vfo
CONOPT	33	0.102	249034.85667
KNITRO	3292	14.310	249034.85667
MINOS	120	0.020	249034.9

Aplicația G15 (*Ramsey*)
Model de creştere economică (echilibru dinamic) Ramsey.
Teoria creşterii economice se bazează pe lucrările de pionierat ale lui Frank Ramsey [1928], care în scurt timp au devenit un instrument standard în modelarea economică aplicată. În domeniul economic Ramsey a publicat două lucrări: *A contribution to the theory of taxation* şi *A mathematical theory of saving.* Acestea au avut un impact deosebit asupra teoriei matematice a economiei, determinând utilizarea modelelor din ce în ce mai sofisticate în acest domeniu.


Frank Ramsey (1903-1930)

Pentru a defini un model de creştere economică să introducem: C_t consumul la momentul de timp t and ρ rata de discaunt. Fie în acelaşi timp $u(C_t)$ o funcție de utilitate instantanee care se bucură de următoarele proprietăți: $u(C_t) \geq 0$, $u'(C_t) > 0$ şi $u''(C_t) < 0$. Cu acestea, maximizarea utilității (preferința consumatorului) conduce la următoarea funcție de bunăstare:

$$\max W = \int_{t=0}^{\infty} u(C_t) e^{-\rho t} \mathrm{d}t.$$

În continuare să introducem $f(K_t)$ funcția de producție dependentă de capitalul K_t, cu următoarele proprietăți: $f(0) = 0$, $f'(0) = \infty$ şi $f'(\infty) = 0$, (Inada conditions [1963]).

Cu acestea ecuația de acumulare a capitalului se defineşte prin intermediul restricției bugetare sub forma:

$$f(K_t) = C_t + \frac{\mathrm{d}K}{\mathrm{d}t},$$

unde termenul $\mathrm{d}K / \mathrm{d}t$ reprezintă investiția netă. Cu alte cuvinte, ieşirea este consumată sau investită.

Studiul analitic al acestui model este prezentat în [Barro şi Martin, 1998], [Barro, 1998, 1999], [Pedersen, 1999]. Pentru studiul comportării numerice vom

introduce o variantă discretă a modelului de mai sus sub forma [Ramsey, 1928], [Kalvelagen, 2003]:

$$\max W = \sum_{t=0}^{\infty} \left(\frac{1}{1+\rho} \right)^t u(C_t)$$

referitor la:

$$Y_t = f(K_t)$$
$$Y_t = C_t + I_t$$
$$K_{t+1} = (1-\delta)K_t + I_t \ ,$$

unde δ este rata de depreciere a capitalului.

De obicei, funcția de utilitate $u(.)$ şi funcția de producție $f(.)$ se definesc sub forma:

$$u(C_t) = \log(C_t),$$
$$f(K_t) = aK_t^b L_t^{1-b},$$

care este o funcție de tip Cobb-Douglas, unde L_t este factorul de producție determinat sub forma:

$$L_{t+1} = (1+g)L_t,$$

unde g este rata de creştere a producției.

Pentru a utiliza acest model este necesar să tratăm orizontul de timp infinit din funcția obiectiv. O soluție, foarte simplă şi la îndemână este da a ignora termenii seriei de la un anumit moment de timp $t = T$. Totuşi, o suluție mai rafinată consideră $C_t = C_T$ pentru $t > T$, adică de la un moment T încolo consumul este constant.

În acest moment să introducem o mică digresiune. După cum ştim pentru orice $0 < a < 1$, putem scrie:

$$\sum_{t=0}^{\infty} a^t = \lim_{T\to\infty} \sum_{t=0}^{T-1} a^t = \lim_{T\to\infty} \frac{1-a^T}{1-a} = \frac{1}{1-a}.$$

Dar,

$$\sum_{t=0}^{T-1} a^t = \frac{1-a^T}{1-a}$$

şi deci,

$$\sum_{t=T}^{\infty} a^t = \sum_{t=0}^{\infty} a^t - \sum_{t=0}^{T-1} a^t = \frac{1}{1-a} - \frac{1-a^T}{1-a} = \frac{a^T}{1-a}.$$

Cu acestea, funcția obiectiv devine;

$$\max W = \sum_{t=0}^{T-1} \left(\frac{1}{1+\rho} \right)^t \log(C_t) + \sum_{t=T}^{\infty} \left(\frac{1}{1+\rho} \right)^t \log(C_T)$$
$$= \sum_{t=0}^{T-1} \left(\frac{1}{1+\rho} \right)^t \log(C_t) + \frac{1}{\rho(1+\rho)^{T-1}} \log(C_T)$$

$$= \sum_{t=0}^{T} \beta_t \log(C_t),$$

unde:

$$\beta_t = \begin{cases} (1+\rho)^{-t} & , \quad t < T, \\ \rho^{-1}(1+\rho)^{1-T}, & \quad t = T. \end{cases}$$

Observăm că astfel formulată, problema asociază o pondere valorii terminale $u(C_T)$. În experimentele de mai jos vom vedea că această pondere este importantă. Mai mult, acest mod de abordare a problemelor de optimizare cu orizont de timp infinit se poate aplica şi în alte contexte.

Trunchierea funcţiei obiectiv impune introducerea unei restricţii la momentul final T referitoare la cerinţa unei investiţii minime cu care să se restarteze mecanismul economic:

$$I_T \geq (g+\delta)K_T,$$

unde, după cum am văzut, g este rata de creştere a producţiei şi δ este rata de depreciere a capitalului.

Cu acestea un model de creştere economică de tip Ramsey se prezintă sub forma:

$$maxW = \sum_{t=0}^{T} \beta_t \log(C_t),$$

referitor la:

$$\begin{aligned} &Y_t = a K_t^b L_t^{1-b}, && t = 0,1,\ldots,T, \\ &Y_t = C_t + I_t, && t = 0,1,\ldots,T, \\ &K_{t+1} = (1-\delta)K_t + I_t, && t = 0,1,\ldots,T-1, \\ &I_T \geq (g+\delta)K_T, \end{aligned}$$

unde ρ (factorul de discaunt), g (rata de creştere a producţiei), δ (factorul de depreciere a capitalului), K_0 (capitalul iniţial), I_0 (investiţia iniţială), C_0 (consumul iniţial), L_0 (factorul de producţie iniţial), b şi a (coeficienţi Cobb-Douglas) sunt parametri cunoscuţi.

Observăm imediat că coeficientul Cobb-Douglas a se poate determina sub forma următoare. Avem: $Y_0 = C_0 + I_0$ şi $Y_0 = f(K_0, L_0) = a K_0^b L_0^{1-b}$, deci

$$a = \frac{C_0 + I_0}{K_0^b L_0^{1-b}}.$$

Expresia acestui model în GAMS [Andrei, 2004b] este prezentată în figura 15.1.

```
$ontext
   Model de crestere economica (echilibru dinamic) Ramsey
$offtext
*
* Date asociate modelului:
*
```

```
set  t          'perioadele de timp'      / t1*t50 /

scalars
      rho       'factorul de discaunt'                    / 0.04 /
      g         'rata de crestere a productiei'           / 0.03 /
      delta     'factorul de depreciere al capitalului'   / 0.02 /
      K0        'capitalul initial'                       / 3.00 /
      I0        'investitia initiala'                     / 0.07 /
      C0        'consumul initial'                        / 0.95 /
      L0        'factorul de productie initial'           / 1.00 /
      b         'coeficient Cobb Douglas'                 / 0.25 /
      a         'coeficient Cobb Douglas' ;
*
* Multimi de lucru
*
sets
      tfirst(t)   'primul interval (t0)'
      tlast(t)    'ultimul intervat (T)'
      tnotlast(t) 'toate intervalele de timp fara ultimul' ;

tfirst(t)$(ord(t)=1) = yes;
tlast(t)$(ord(t)=card(t)) = yes;
tnotlast(t)= not tlast(t);
*
parameters
      L(t)      'factorul de productie'
      beta(t)   'ponderea utilitatilor in functia obiectiv'
      tval(t)   'valoarea numerica a lui t'  ;

tval(t) = ord(t)-1;
*
* Calculul ponderilor utilitatilor.
* Ponderea asociata ultimului interval de timp (ponderea terminala)
* compenseaza folosirea utilitatilor dincolo de momentul de timp T.
*
beta(tnotlast(t)) = power(1+rho,-tval(t));
beta(tlast(t)) = (1/rho)*power(1+rho,1-tval(t));
display beta;
*
* Calculul factorului de productie utilizand un proces de crestere
* exponentiala.
*
L(t) = power(1+g,tval(t)) * L0;
*
* Calculul coeficientului Cobb-Douglas a.
*
a = (C0 + I0) / (K0**b * L0**(1-b));
*
variables
      C(t)      'consumul'
      Y(t)      'productia'
      K(t)      'capitalul'
      I(t)      'investitia'
      W         'utilitatea totala'  ;
*
equation
      utilitate         'functia obiectiv, utilitatea'
      productie(t)      'functia de productie Cobb-Douglas'
      alocare(t)        'ecuatia productiei'
      acumulare(t)      'ecuatia acumularii capitalului'
      final(t)          'investitia in ultimul intervat'  ;

utilitate..         W =e= sum(t, beta(t)*log(C(t)));
```

```
productie(t)..       Y(t) =e= a * (K(t)**b) * (L(t)**(1-b));
alocare(t)..         Y(t) =e= C(t) + I(t);
acumulare(tnotlast(t))..  K(t+1) =e= (1-delta)*K(t) + I(t);
final(tlast)..       I(tlast) =g= (g+delta)*K(tlast);
*
* Margini asupra unor variabile pentru a putea evalua functiile.
*
K.lo(t) = 0.001;
C.lo(t) = 0.001;
*
* Conditiile initiale.
*
K.fx(tfirst) = K0;
I.fx(tfirst) = I0;
C.fx(tfirst) = C0;

model ramsey /all/;
solve ramsey maximizing W using nlp;
*
* Afisarea solutiei
*
file res1 /growth.txt/
put res1;
put /"timp    C(t)  Y(t)  K(t)  I(t) "/;
loop(t, put t.tl:6, C.l(t):6, Y.l(t):6, K.l(t):6, I.l(t):6 /;);

display beta
* End Ramsey
```

Fig. 15.1. Expresia GAMS a aplicației G15 (Ramsey).

Soluția modelului, corespunzătoare datelor precizate este arătată în figura 15.2

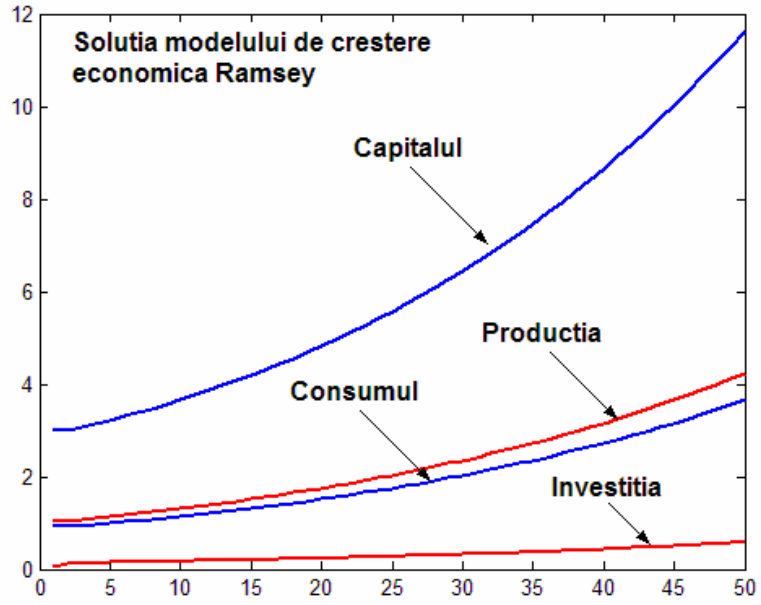

Fig. 15.2. Soluția modelului Ramsey.

O variantă de model Ramsey cu 30 de variabile şi 21 de restricții este prezentată în [Andrei, 2003, pag. 398], unde se prezintă şi soluția acestuia obținută cu pachetul SPENBAR [Andrei, 1996a,b].

Modele dinamice de tip Ramsey sunt foarte des utilizate în contextul modelării economice. Putem menționa modelul *DICE - Dynamic Integrated Climate-Economy* dezvoltat de Nordhaus [1992]; modelul *TAVF - Transitional Alternative Fuels and Vehicles* prezentat de Leiby şi Rubin [1997] şi *MEEN - Modeling the Environment-Economy Nexus* al lui Harrison [2001]. O clasă specială de modele bazată pe teoria modelelor dinamice Ramsey este dată de modelele *MARKAL-MACRO* [Manne şi Wene, 1992].

Expresia GAMS a modelului conține instrucțiunea de afişare a ponderilor β_t din funcția obiectiv (display beta;). Valorile acestor ponderi sunt următoarele:

```
----    119 PARAMETER beta  ponderea utilitatilor in functia obiectiv

t1  1.000,    t2  0.962,    t3  0.925,    t4  0.889,    t5  0.855,    t6  0.822
t7  0.790,    t8  0.760,    t9  0.731,    t10 0.703,    t11 0.676,    t12 0.650
t13 0.625,    t14 0.601,    t15 0.577,    t16 0.555,    t17 0.534,    t18 0.513
t19 0.494,    t20 0.475,    t21 0.456,    t22 0.439,    t23 0.422,    t24 0.406
t25 0.390,    t26 0.375,    t27 0.361,    t28 0.347,    t29 0.333,    t30 0.321
t31 0.308,    t32 0.296,    t33 0.285,    t34 0.274,    t35 0.264,    t36 0.253
t37 0.244,    t38 0.234,    t39 0.225,    t40 0.217,    t41 0.208,    t42 0.200
t43 0.193,    t44 0.185,    t45 0.178,    t46 0.171,    t47 0.165,    t48 0.158
t49 0.152,    t50 3.805
```

Observăm că ponderea asociată momentului final β_{50} are o valoare substanțială. Aceasta arată încă odată importanța tehnicii de transformare a seriei infinite din funcția obiectiv într-o sumă finită.

Performanțele algoritmilor CONOPT, KNITRO şi MINOS sunt prezentate în tabelul 15.1.

Tabelul 15.1

$n = 201$ (variabile), $m = 151$ (restricții)

	#iter	time	Vfo
CONOPT	40	0.168	12.797918
KNITRO	11	0.080	12.797918
MINOS	18	0.223	12.797920

Aplicația G16 (*Catmix*)

Determinarea amestecului optim a două elemente catalizatoare de-a lungul unui reactor tubular implicând mai mulți reactanți.

Modelul matematic care descrie reacțiile chimice este descris în [von Stryk, 1999] şi [Dolan, Moré şi Munson, 2004] are forma:

$$x_1'(t) = u(t)(10x_2(t) - x_1(t)),$$
$$x_2'(t) = u(t)(x_1(t) - 10x_2(t)) - (1 - u(t))x_2(t).$$

Condițiile inițiale sunt $x_1(0)=1$ şi $x_2(t)=0$. Variabila de control $u(t)$ reprezintă raportul dintre cele două substanțe catalizatoare şi satisface condiția de mărginire

$$0 \le u(t) \le 1.$$

Problema este de a minimiza

$$-1 + x_1(t_f) + x_2(t_f), \quad t_f = 1.$$

Discretizarea problemei constă în împărțirea uniformă a intervalului [0,1] în n_h subintervale şi reformularea ecuațiilor utilizând metoda colocației în k etape utilizând reprezentarea standard a bazei. Ca punct inițial considerăm $u=0$, $x_1=1$ şi $x_2=0$ evaluate în punctele de discretizare. Restricțiile problemei sunt date de condițiile inițlae, ecuațiile de continuitate şi ecuațiile de colocație. Expresia GAMS a acestei aplicații este dată în figura 16.1.

```
$ontext
Determine the optimal policy of two catalysts along the length
of a tubular plug flow reactor involving several reactions.
$offtext

$if     set n  $set nh %n%
$if not set nh $set nh 800

Set nh Number of subintervals / 0*%nh% /;
Alias (nh,i);

Scalar tf Final time                  /1/
       x1_0 Initial condition for x1  /1/
       x2_0 Initial condition for x2  /0/
       alpha smoothing parameter      /0/
       h;

       h = tf/%nh%;

Variable u(nh), x1(nh), x2(nh), obj;

Positive variable u; u.up(nh) = 1;

Equations defobj objective function
          ode1(nh)
          ode2(nh);

defobj.. obj =e= -1 + x1['%nh%'] + x2['%nh%'] +
                  alpha*h*sum{nh(i+1), sqr(u[i+1] - u[i])};

ode1(nh(i+1)).. x1[i+1] =e= x1[i] +
                  (h/2)*(u[i]*(10*x2[i]-x1[i]) +
                         u[i+1]*(10*x2[i+1]-x1[i+1]));

ode2(nh(i+1)).. x2[i+1] =e= x2[i] +
                  (h/2)*(u[i]*(x1[i]-10*x2[i]) -
                        (1-u[i])*x2[i] +
                         u[i+1]*(x1[i+1]-10*x2[i+1]) -
                        (1-u[i+1])*x2[i+1]);
```

```
* Initial point
x1.l[nh] = 1;

x1.fx['0'] = x1_0;
x2.fx['0'] = x2_0;

model catmix /all/;

$onecho >bench.opt
  solvers conopt knitro minos
$offecho
catmix.optfile=1;

option nlp=bench;

solve catmix minimizing obj using nlp;

*file res1 /catmix.dat/;
*put res1
*loop(nh, put x1.l(nh):10:5, put/)

* End catmix
```

Fig. 16.1. Expresia GAMS a aplicației G16 (Catmix).

Performanțele algoritmilor CONOPT, KNITRO şi MINOS sunt prezentate în tabelele de mai jos

Tabelul 16.1

$n_h = 400,\ n = 1204$ (variabile), $m = 801$ (restricții)

	#iter	time	vfo
CONOPT	60	0.469	-0.04805651
KNITRO	12	0.280	-0.04800083
MINOS	8	0.719	-0.04805472

Tabelul 16.2

$n_h = 800,\ n = 2404$ (variabile), $m = 1601$ (restricții)

	#iter	time	vfo
CONOPT	88	1.523	-0.04805586
KNITRO	12	0.490	-0.04795030
MINOS	10	1.969	-0.04805527

Tabelul 16.3

$n_h = 1000,\ n = 3004$ (variabile), $m = 2001$ (restricții)

	#iter	time	vfo
CONOPT	82	1.820	-0.04805574
KNITRO	12	0.610	-0.04792625
MINOS	11	3.016	-0.04805545

Tabelul 16.4

$n_h = 2000,\ \ n = 6004$ (variabile), $m = 4001$ (restricții)

	#iter	time	vfo
CONOPT	81	8.047	-0.04805526
KNITRO	11	1.782	-0.04781198
MINOS	18	14.656	-0.0480529

Tabelul 16.5

$n_h = 3000,\ \ n = 9004$ (variabile), $m = 6001$ (restricții)

	#iter	time	vfo
CONOPT	33	3.922	-0.04805360
KNITRO	8	2.223	-0.04795616
MINOS	23	30.664	-0.04804791

Figura 16.2 prezintă evoluția variabilei $u(t)$ dată de CONOPT şi respectiv MINOS. Vedem caracterul de bang-bang singular al problemei impus de prezența restricțiilor de mărginire asupra lui $u(t)$.

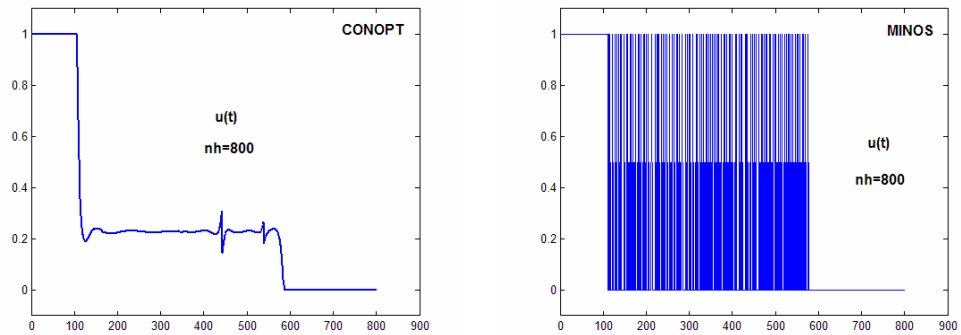

Fig. 16.2. Evoluția variabilei $u(t)$ dată de CONOPT şi MINOS.

Figura 16.3 prezintă evoluția variabilelor $x_1(t)$, respectiv $x_2(t)$ dată de CONOPT.

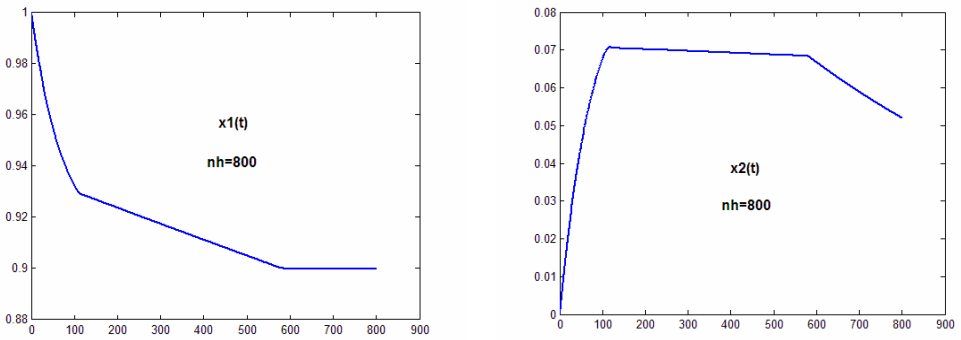

Fig. 16.3. Evoluția variabilelor $x_1(t)$ şi $x_2(t)$ dată de CONOPT.

Aplicația G17 (*Pool1*)

Pooling – Blending. Determinarea în mai multe tancuri a unui amestec din mai multe ingrediente cu proprietăți date în vederea obținerii unor produse finite cu proprietăți impuse (4 ingrediente, un tanc, 2 produse finite).

Problemele de acest tip, cunoscute ca pooling şi blending sunt comune în industria chimică şi petro-chimică. Dae mai multe ingrediente cu diferite proprietăți chimice şi costuri, scopul este de a se determina debitele optime ale acestora în anumite tancuri care determină anumite produse chimice finite cu proprietăți şi costuri cunoscute. Ingredientele se pot amesteca în tancuri, sau se pot introduce direct în produsele finite. Pentru fiecare dintre aceste elemente, ingrediente, tancuri şi produse finite, se cunosc o serie de caracteristici care definesc o situație fizică concretă. Aceste probleme au fost studiate, între alții, de Haverly [1978], Lasdon *et al*. [1979], Ben-Tal *et al.* [1994] , Visweswaran şi Floudas [1996], Andrei [2001].

În figura 17.1 se prezintă o situație generală de amestec pentru care se prezintă modelul matematic. Este vorba despre un sistem de amestec care conține trei ingrediente, un tanc şi două produse finite [Floudas *et al.*, 1999].

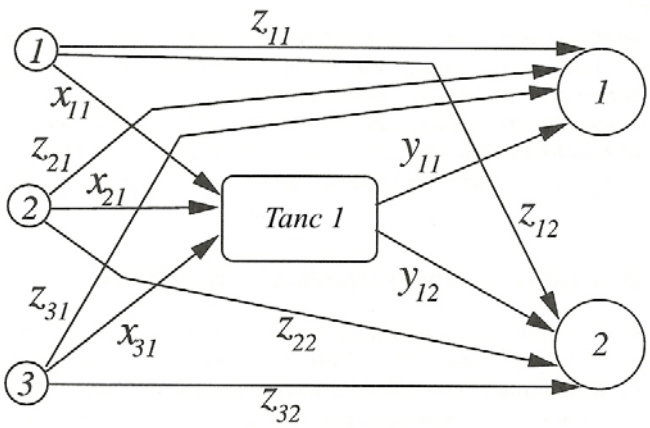

Fig. 17.1. Schema generală a unui sistem de amestec.

Vectorii q, y şi z reprezintă debitele între diferite unități ale sistemului de amestec. În acest sistem q_{il} reprezintă debitul din ingredientul i care ajunge în tancul l. y_{lj} este debitul de la tancul l care ajunge la produsul finit j, iar z_{ij} este debitul din ingredientul i care ajunge direct la produsul finit j. Cu acestea definim următoarele mulțimi de indici. I mulțimea ingredientelor, J mulțimea produselor finite, L mulțimea tancurilor şi K mulțimea acelor componente a căror calitate trebuie monitorizată. Pentru sistemul de amestec introducem următorii parametri care definesc o sitiație fizică concretă. A_i este debitul maxim care poate curge de la ingredientul i. D_j este cererea maximă din produsul finit (care se vinde) j. S_l este capacitatea maximă a tancului l. C_{ik} este procentul din componenta k în ingredientul i. P_{jk} este procentul maxim din componenta k în produsul finit j. c_i

este preţul unitar al ingredientului i şi d_j este preţul unitar a produsului finit j. Cu acestea modelul matematic al sistemului de amestec este următorul.

$$\max_{q,y,z} \sum_{j\in J}\sum_{l\in L}(d_j - \sum_{i\in I} c_i q_{il}) y_{lj} + \sum_{i\in I}\sum_{j\in J}(d_j - c_i) z_{ij},$$

referitor la:

$$\sum_{l\in L}\sum_{j\in J} q_{il} y_{lj} + \sum_{j\in J} z_{ij} \le A_i, \quad \forall i \in I,$$

$$\sum_{j\in J} y_{lj} \le S_l, \quad \forall l \in L,$$

$$\sum_{l\in L} y_{lj} + \sum_{i\in I} z_{ij} \le D_j, \quad \forall j \in J,$$

$$\sum_{l\in L}\left(\sum_{i\in I} C_{ik} q_{il} - P_{jk}\right) y_{lj} + \sum_{i\in I}(C_{ik} - P_{jk}) z_{ij} \le 0, \quad \forall j \in J, \ \forall k \in K,$$

$$\sum_{i\in I} q_{il} = 1, \quad \forall l \in L.$$

Modelul se completează cu margini asupra variabilelor:

$$0 \le q_{il} \le 1, \quad \forall i \in I, \quad \forall l \in L,$$

$$0 \le y_{lj} \le D_j, \quad \forall l \in L, \quad \forall j \in J,$$

$$0 \le z_{ij} \le D_j, \quad \forall i \in I, \quad \forall j \in J.$$

După cum se vede modelul are o formă biliniară, şi formează o subclasă importantă de probleme de optimizare pătratică neconvexe. Majoritatea algoritmilor întâmpină greutăţi considerabile în rezolvarea acestor probleme şi după cum vom vedea diferiţi algoritmi furnizează soluţii diferite.

Să considerăm o situaţie de sistem de amestec cu 4 ingrediente, un tanc şi două produse finite, ca în figura 17.2 [Floudas *et al.*, 1999].

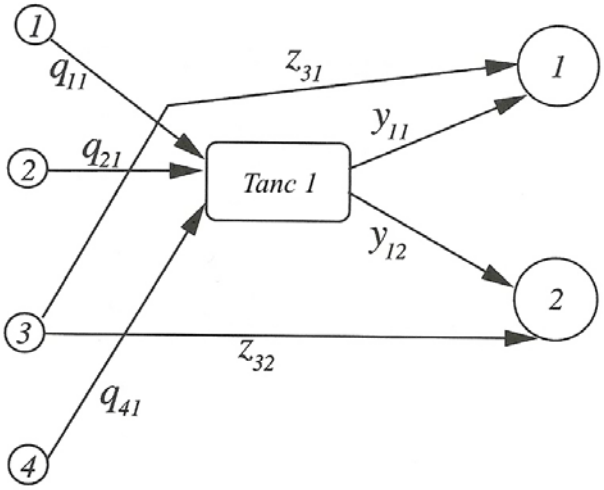

Fig. 17.2. Schema amestecului corespunzător aplicaţiei G17.

Modelul matematic al acestui sistem de ameste este următorul:

$$\max(9-6q_{11}-16q_{21}-15q_{41})y_{11}+$$
$$(15-16q_{11}-16q_{21}-15q_{41})y_{12}-z_{31}+5z_{32}$$

referitor la:

$$q_{41}y_{11}+q_{41}y_{12}\le 50,$$
$$y_{11}+z_{31}\le 100,$$
$$y_{12}+z_{32}\le 200,$$
$$(3q_{11}+q_{21}+q_{41}-2.5)y_{11}-0.5z_{31}\le 0,$$
$$(3q_{11}+q_{21}+q_{41}-2.5)y_{12}-0.5z_{32}\le 0,$$
$$q_{11}+q_{21}+q_{41}=1.$$

Marginile asupra variabilelor sunt:

$$0\le q_{11}\le 1,\quad 0\le q_{21}\le 1,\quad 0\le q_{41}\le 1,$$
$$0\le y_{11}\le 100,\quad 0\le y_{12}\le 200,$$
$$0\le z_{31}\le 100,\quad 0\le z_{32}\le 200.$$

După cum se vede datele acestei probleme sunt:

$$A=(\infty,\infty,\infty,50),\quad D=(100,200),\quad S_1=\infty,$$
$$C=(3,1,-,1),\quad P=(2.5,1.5),$$
$$c=(6,16,-,15),\quad d=(9,15).$$

Expresia GAMS a acestei aplicații este arătată în figura 17.3.

```
$ontext
Determinarea amestecului optim format din 4 ingrediente, un
tanc si 2 produse finite.
$offtext
VARIABLES
         q11  fraction of flow to pool 1 coming from feed 1
         q21  fraction of flow to pool 1 coming from feed 2
         q41  fraction of flow to pool 1 coming from feed 4
         y11  flow from pool 1 to product 1
         y12  flow from pool 1 to product 2
         z31  flow from feed 3 to product 1
         z32  flow from feed 3 to product 2
      objval  objective function variable;

FREE VARIABLES    objval;


EQUATIONS
         f Objective function
         g1, g2, g3, g4, g5
         g6  ;

f  .. objval =e= (9 - 6*q11 - 16*q21 - 15*q41)*y11 +
                 (15 - 6*q11 - 16*q21 - 15*q41)*y12 -
                 z31 + 5*z32;
```

```
g1  .. q41*y11 + q41*y12 =l= 50;
g2  .. y11 + z31 =l= 100;
g3  .. y12 + z32 =l= 200;
g4  .. (3*q11 + q21 + q41 -2.5)*y11 - 0.5*z31 =l= 0;
g5  .. (3*q11 + q21 + q41 -1.5)*y12 + 0.5*z32 =l= 0;
g6  .. q11 + q21 + q41 =e= 1;

* Bounds
q11.LO = 0; q11.UP = 1;
q21.LO = 0; q21.UP = 1;
q41.LO = 0; q41.UP = 1;
y12.LO = 0; y12.UP = 200;
y11.LO = 0; y11.UP = 100;
z31.LO = 0; z31.UP = 100;
z32.LO = 0; z32.UP = 200;

* Starting point (global solution)
* q11.L =  0; q21.L =  0.5; q41.L =  0.5;
* y11.L =  0; y12.L =  100;
* z31.L =  0; z32.L =  100;

MODEL pool1 /ALL/;
$onecho >bench.opt
  solvers conopt knitro minos
$offecho
pool1.optfile=1;
option nlp=bench;
SOLVE pool1 USING NLP MAXIMIZING objval;
* End Pool1
```

Fig. 17.3. Expresia GAMS a aplicației G17 (Pool1).

Soluția acestei probleme şi caracteristicile procesului de optimizare cu CONOPT, KNITRO şi MINOS este următoarea:

	CONOPT	**KNITRO**	**MINOS**
q_{11}	0.333	0	1
q_{21}	0.333	0.5	0
q_{41}	0.333	0.5	0
y_{11}	0	0	50
y_{12}	0	100	0
z_{31}	0	0	50
z_{32}	0	100	0
#iter	3	28	6
time	0.008	0.190	0.098
vfo	0	450	100

Observăm diferențele majore care există între soluțiile furnizate de algoritmii considerați. Numai KNITRO este în stare de a furniza o soluție pentru care valoarea funcției obiectv, vfo, este maximă.

Aplicația G18 (*Pool2*)

Pooling – Blending. Determinarea în mai multe tancuri a unui amestec din mai multe ingrediente cu proprietăți date în vederea obținerii unor produse finite cu proprietăți impuse (5 ingrediente, 3 tancuri, 5 produse finite).

Să considerăm de un sistem de amestec în care este vorba de 5 ingrediente care se amestecă în 3 tancuri din care se livrează 5 produse finite, figura 18.1 [Floudas *et al.*, 1999].

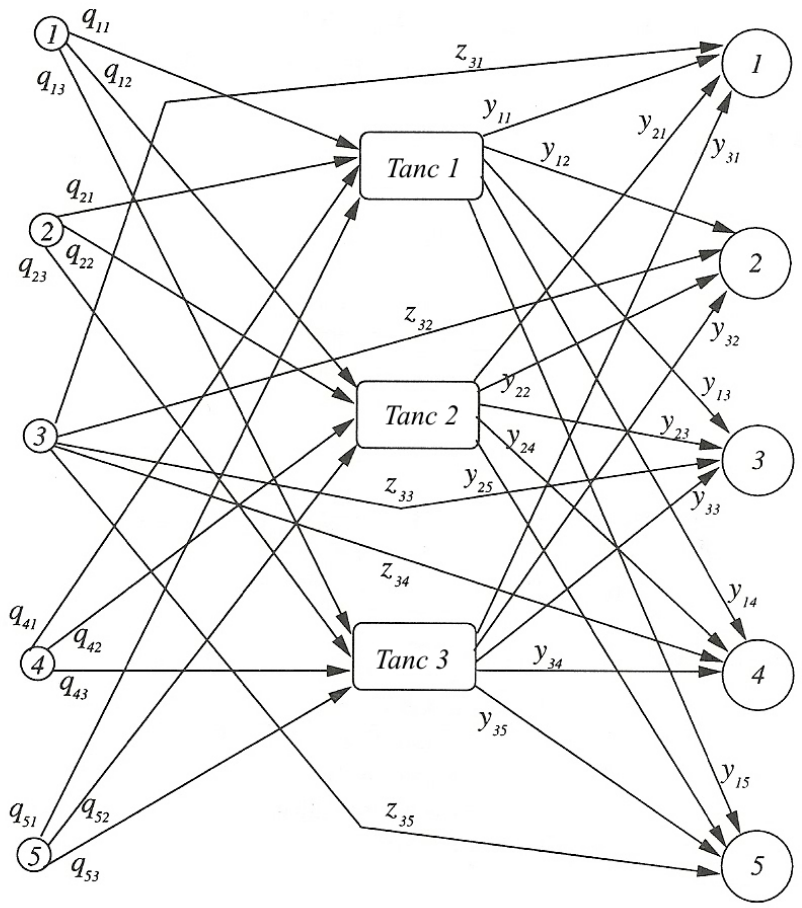

Fig. 18.1. Schema amestecului corespunzător aplicației G18.

Modelul matematic corespunzător acestei scheme de amestec este următorul:

$$
\begin{aligned}
\max\ & (18-6q_{11}-16q_{21}-15q_{41}-12q_{51})y_{11}+\\
& (18-6q_{12}-16q_{22}-15q_{42}-12q_{52})y_{21}+\\
& (18-6q_{13}-16q_{23}-15q_{43}-12q_{53})y_{31}+\\
& (15-6q_{11}-16q_{21}-15q_{41}-12q_{51})y_{12}+\\
& (15-6q_{12}-16q_{22}-15q_{42}-12q_{52})y_{22}+
\end{aligned}
$$

$$(15-6q_{13}-16q_{23}-15q_{43}-12q_{53})y_{32}+$$
$$(19-6q_{11}-16q_{21}-15q_{41}-12q_{51})y_{13}+$$
$$(19-6q_{12}-16q_{22}-15q_{42}-12q_{52})y_{23}+$$
$$(19-6q_{13}-16q_{23}-15q_{43}-12q_{53})y_{33}+$$
$$(16-6q_{11}-16q_{21}-15q_{41}-12q_{51})y_{14}+$$
$$(16-6q_{12}-16q_{22}-15q_{42}-12q_{52})y_{24}+$$
$$(16-6q_{13}-16q_{23}-15q_{43}-12q_{53})y_{34}+$$
$$(14-6q_{11}-16q_{21}-15q_{41}-12q_{51})y_{15}+$$
$$(14-6q_{12}-16q_{22}-15q_{42}-12q_{52})y_{25}+$$
$$(14-6q_{13}-16q_{23}-15q_{43}-12q_{53})y_{35}+$$
$$8z_{31}+5z_{32}+9z_{33}+6z_{34}+4z_{35},$$

refereitor la:

$$q_{41}(y_{11}+y_{12}+y_{13}+y_{14}+y_{15})+q_{42}(y_{21}+y_{22}+y_{23}+y_{24}+y_{25})+$$
$$q_{43}(y_{31}+y_{32}+y_{33}+y_{34}+y_{35})\le 50,$$
$$y_{11}+y_{21}+y_{31}+z_{31}\le 100,$$
$$y_{12}+y_{22}+y_{32}+z_{32}\le 200,$$
$$y_{13}+y_{23}+y_{33}+z_{33}\le 100,$$
$$y_{14}+y_{24}+y_{34}+z_{34}\le 100,$$
$$y_{15}+y_{25}+y_{35}+z_{35}\le 100,$$
$$(3q_{11}+q_{21}+q_{41}+1.5q_{51}-2.5)y_{11}+(3q_{12}+q_{22}+q_{42}+1.5q_{52}-2.5)y_{21}+$$
$$(3q_{13}+q_{23}+q_{43}+1.5q_{53}-2.5)y_{31}-0.5z_{31}\le 0,$$
$$(q_{11}+3q_{21}+2.5q_{41}+2.5q_{51}-2)y_{11}+(q_{12}+3q_{22}+2.5q_{42}+2.5q_{52}-2)y_{21}+$$
$$(q_{13}+3q_{23}+2.5q_{43}+2.5q_{53}-2)y_{31}+0.5z_{31}\le 0,$$
$$(3q_{11}+q_{21}+q_{41}+1.5q_{51}-1.5)y_{12}+(3q_{12}+q_{22}+q_{42}+1.5q_{52}-1.5)y_{22}+$$
$$(3q_{13}+q_{23}+q_{43}+1.5q_{53}-1.5)y_{32}-0.5z_{32}\le 0,$$
$$(q_{11}+3q_{21}+2.5q_{41}+2.5q_{51}-2.5)y_{12}+(q_{12}+3q_{22}+2.5q_{42}+2.5q_{52}-2.5)y_{22}+$$
$$(q_{13}+3q_{23}+2.5q_{43}+2.5q_{53}-2.5)y_{32}\le 0,$$
$$(3q_{11}+q_{21}+q_{41}+1.5q_{51}-2)y_{13}+(3q_{12}+q_{22}+q_{42}+1.5q_{52}-2)y_{23}+$$
$$(3q_{13}+q_{23}+q_{43}+1.5q_{53}-2)y_{33}\le 0,$$
$$(q_{11}+3q_{21}+2.5q_{41}+2.5q_{51}-2.6)y_{13}+(q_{12}+3q_{22}+2.5q_{42}+2.5q_{52}-2.6)y_{23}+$$
$$(q_{13}+3q_{23}+2.5q_{43}+2.5q_{53}-2.6)y_{33}-0.1z_{33}\le 0,$$
$$(3q_{11}+q_{21}+q_{41}+1.5q_{51}-2)y_{14}+(3q_{12}+q_{22}+q_{42}+1.5q_{52}-2)y_{24}+$$
$$(3q_{13}+q_{23}+q_{43}+1.5q_{53}-2)y_{34}\le 0,$$
$$(q_{11}+3q_{21}+2.5q_{41}+2.5q_{51}-2)y_{14}+(q_{12}+3q_{22}+2.5q_{42}+2.5q_{52}-2)y_{24}+$$
$$(q_{13}+3q_{23}+2.5q_{43}+2.5q_{53}-2)y_{34}+0.5z_{34}\le 0,$$
$$(3q_{11}+q_{21}+q_{41}+1.5q_{51}-2)y_{15}+(3q_{12}+q_{22}+q_{42}+1.5q_{52}-2)y_{25}+$$

$$(3q_{13}+q_{23}+q_{43}+1.5q_{53}-2)y_{35}\le 0,$$
$$(q_{11}+3q_{21}+2.5q_{41}+2.5q_{51}-2)y_{15}+(q_{12}+3q_{22}+2.5q_{42}+2.5q_{52}-2)y_{25}+$$
$$(q_{13}+3q_{23}+2.5q_{43}+2.5q_{53}-2)y_{35}+0.5z_{35}\le 0,$$
$$q_{11}+q_{21}+q_{41}+q_{51}=1,$$
$$q_{12}+q_{22}+q_{42}+q_{52}=1,$$
$$q_{13}+q_{23}+q_{43}+q_{53}=1.$$

Marginile asupra variabilelor sunt:

$$\begin{array}{lll}
0\le q_{11}\le 1, & 0\le q_{12}\le 1, & 0\le q_{13}\le 1,\\
0\le q_{21}\le 1, & 0\le q_{22}\le 1, & 0\le q_{23}\le 1,\\
0\le q_{41}\le 1, & 0\le q_{42}\le 1, & 0\le q_{43}\le 1,\\
0\le q_{51}\le 1, & 0\le q_{52}\le 1, & 0\le q_{53}\le 1,\\
0\le y_{11}\le 100, & 0\le y_{21}\le 100, & 0\le y_{31}\le 100,\\
0\le y_{12}\le 200, & 0\le y_{22}\le 200, & 0\le y_{32}\le 200,\\
0\le y_{13}\le 100, & 0\le y_{23}\le 100, & 0\le y_{33}\le 100,\\
0\le y_{14}\le 100, & 0\le y_{24}\le 100, & 0\le y_{34}\le 100,\\
0\le y_{15}\le 100, & 0\le y_{25}\le 100, & 0\le y_{35}\le 100,\\
0\le z_{31}\le 100, & 0\le z_{32}\le 200, & 0\le z_{33}\le 100,\\
0\le z_{34}\le 100, & 0\le z_{35}\le 100. &
\end{array}$$

Pentru această aplicație datele sunt următoarele:

$$A=(\infty,\infty,\infty,50,\infty),\quad D=(100,200,100,100,100),\quad S=(\infty,\infty,\infty),$$
$$c=(6,16,-,15,12),\quad d=(18,15,19,16,14),$$
$$C=\begin{bmatrix}3.0 & 1.0\\ 1.0 & 3.0\\ 2.0 & 2.5\\ 1.5 & 2.5\end{bmatrix},\quad P=\begin{bmatrix}2.5 & 2.0\\ 1.5 & 2.5\\ 2.0 & 2.6\\ 2.0 & 2.0\\ 2.0 & 2.0\end{bmatrix}.$$

Expresia GAMS a acestei aplicații este prezentată în figura 18.2.

```
$ontext
Determinarea  amestecului  optim  format  din  5  ingrediente,  3
tancuri si 5 produse finite
$offtext
VARIABLES
         q11  fraction of flow to pool 1 from feed 1
         q12  fraction of flow to pool 1 from feed 2
         q13  fraction of flow to pool 1 from feed 3
         q21  fraction of flow to pool 2 from feed 1
         q22  fraction of flow to pool 2 from feed 2
         q23  fraction of flow to pool 2 from feed 3
         q41  fraction of flow to pool 4 from feed 1
```

```
         q42  fraction of flow to pool 4 from feed 2
         q43  fraction of flow to pool 4 from feed 3
         q51  fraction of flow to pool 5 from feed 1
         q52  fraction of flow to pool 5 from feed 2
         q53  fraction of flow to pool 5 from feed 3
         z31  flow from feed 3 to product 1
         z32  flow from feed 3 to product 2
         z33  flow from feed 3 to product 3
         z34  flow from feed 3 to product 4
         z35  flow from feed 3 to product 5
         y11  flow from pool 1 to product 1
         y12  flow from pool 1 to product 2
         y13  flow from pool 1 to product 3
         y14  flow from pool 1 to product 4
         y15  flow from pool 1 to product 5
         y21  flow from pool 2 to product 1
         y22  flow from pool 2 to product 2
         y23  flow from pool 2 to product 3
         y24  flow from pool 2 to product 4
         y25  flow from pool 2 to product 5
         y31  flow from pool 3 to product 1
         y32  flow from pool 3 to product 2
         y33  flow from pool 3 to product 3
         y34  flow from pool 3 to product 4
         y35  flow from pool 3 to product 5
      objval  objective function variable;

FREE VARIABLES    objval;

EQUATIONS
         f Objective function
         g1,  g2,  g3,  g4,  g5,  g6,  g7,  g8,  g9, g10
         g11, g12, g13, g14, g15, g16, g17, g18
         g19  ;

f  .. objval =e= (18 - 6*q11 - 16*q21 - 15*q41 - 12*q51)*y11 +
                 (18 - 6*q12 - 16*q22 - 15*q42 - 12*q52)*y21 +
                 (18 - 6*q13 - 16*q23 - 15*q43 - 12*q53)*y31 +
                 (15 - 6*q11 - 16*q21 - 15*q41 - 12*q51)*y12 +
                 (15 - 6*q12 - 16*q22 - 15*q42 - 12*q52)*y22 +
                 (15 - 6*q13 - 16*q23 - 15*q43 - 12*q53)*y32 +
                 (19 - 6*q11 - 16*q21 - 15*q41 - 12*q51)*y13 +
                 (19 - 6*q12 - 16*q22 - 15*q42 - 12*q52)*y23 +
                 (19 - 6*q13 - 16*q23 - 15*q43 - 12*q53)*y33 +
                 (16 - 6*q11 - 16*q21 - 15*q41 - 12*q51)*y14 +
                 (16 - 6*q12 - 16*q22 - 15*q42 - 12*q52)*y24 +
                 (16 - 6*q13 - 16*q23 - 15*q43 - 12*q53)*y34 +
                 (14 - 6*q11 - 16*q21 - 15*q41 - 12*q51)*y15 +
                 (14 - 6*q12 - 16*q22 - 15*q42 - 12*q52)*y25 +
                 (14 - 6*q13 - 16*q23 - 15*q43 - 12*q53)*y35 +
                 8*z31 + 5*z32 + 9*z33 + 6*z34 + 4*z35;

g1  .. q41*y11 + q41*y12 + q41*y13 + q41*y14 + q41*y15 +
       q42*y21 + q42*y22 + q42*y23 + q42*y24 + q42*y25 +
       q43*y31 + q43*y32 + q43*y33 + q43*y34 + q43*y35 =l= 50;
```

```
g2  .. y11 + y21 + y31 + z31 =l= 100;
g3  .. y12 + y22 + y32 + z32 =l= 200;
g4  .. y13 + y23 + y33 + z33 =l= 100;
g5  .. y14 + y24 + y34 + z34 =l= 100;
g6  .. y15 + y25 + y35 + z35 =l= 100;

g7  .. (3*q11 + q21 + q41 + 1.5*q51 -2.5)*y11 +
       (3*q12 + q22 + q42 + 1.5*q52 - 2.5)*y21 +
       (3*q13 + q23 + q43 + 1.5*q53 - 2.5)*y31 - 0.5*z31 =l= 0;

g8  .. (q11 + 3*q21 + 2.5*q41 + 2.5*q51 -2)*y11 +
       (q12 + 3*q22 + 2.5*q42 + 2.5*q52 - 2)*y21 +
       (q13 + 3*q23 + 2.5*q43 + 2.5*q53 - 2)*y31 + 0.5*z31
        =l= 0;
g9  .. (3*q11 + q21 + q41 + 1.5*q51 -1.5)*y12 +
       (3*q12 + q22 + q42 + 1.5*q52 - 1.5)*y22 +
       (3*q13 + q23 + q43 + 1.5*q53 - 1.5)*y32 + 0.5*z32
        =l= 0;
g10  .. (q11 + 3*q21 + 2.5*q41 + 2.5*q51 -2.5)*y12 +
        (q12 + 3*q22 + 2.5*q42 + 2.5*q52 - 2.5)*y22 +
        (q13 + 3*q23 + 2.5*q43 + 2.5*q53 - 2.5)*y32 =l= 0;
g11  .. (3*q11 + q21 + q41 + 1.5*q51 -2)*y13 +
        (3*q12 + q22 + q42 + 1.5*q52 - 2)*y23 +
        (3*q13 + q23 + q43 + 1.5*q53 - 2)*y33 =l= 0;
g12  .. (q11 + 3*q21 + 2.5*q41 + 2.5*q51 -2.6)*y13 +
        (q12 + 3*q22 + 2.5*q42 + 2.5*q52 - 2.6)*y23 +
        (q13 + 3*q23 + 2.5*q43 + 2.5*q53 - 2.6)*y33 - 0.1*z33
         =l= 0;
g13  .. (3*q11 + q21 + q41 + 1.5*q51 -2)*y14 +
        (3*q12 + q22 + q42 + 1.5*q52 - 2)*y24 +
        (3*q13 + q23 + q43 + 1.5*q53 - 2)*y34 =l= 0;
g14  .. (q11 + 3*q21 + 2.5*q41 + 2.5*q51 -2)*y14 +
        (q12 + 3*q22 + 2.5*q42 + 2.5*q52 - 2)*y24 +
        (q13 + 3*q23 + 2.5*q43 + 2.5*q53 - 2)*y34 + 0.5*z34
         =l= 0;
g15  .. (3*q11 + q21 + q41 + 1.5*q51 -2)*y15 +
        (3*q12 + q22 + q42 + 1.5*q52 - 2)*y25 +
        (3*q13 + q23 + q43 + 1.5*q53 - 2)*y35  =l= 0;
g16  .. (q11 + 3*q21 + 2.5*q41 + 2.5*q51 -2)*y15 +
        (q12 + 3*q22 + 2.5*q42 + 2.5*q52 - 2)*y25 +
        (q13 + 3*q23 + 2.5*q43 + 2.5*q53 - 2)*y35 + 0.5*z35
         =l= 0;

g17  .. q11 + q21 + q41 + q51 =e= 1;
g18  .. q12 + q22 + q42 + q52 =e= 1;
g19  .. q13 + q23 + q43 + q53 =e= 1;

* Bounds
q11.LO = 0; q11.UP = 1;
q12.LO = 0; q12.UP = 1;
q13.LO = 0; q13.UP = 1;
q21.LO = 0; q21.UP = 1;
q22.LO = 0; q22.UP = 1;
q23.LO = 0; q23.UP = 1;
q41.LO = 0; q41.UP = 1;
q42.LO = 0; q42.UP = 1;
```

```
q43.LO = 0; q43.UP = 1;
q51.LO = 0; q51.UP = 1;
q52.LO = 0; q52.UP = 1;
q53.LO = 0; q53.UP = 1;
y11.LO = 0; y11.UP = 100;
y12.LO = 0; y12.UP = 200;
y13.LO = 0; y13.UP = 100;
y14.LO = 0; y14.UP = 100;
y15.LO = 0; y15.UP = 100;
y21.LO = 0; y21.UP = 100;
y22.LO = 0; y22.UP = 200;
y23.LO = 0; y23.UP = 100;
y24.LO = 0; y24.UP = 100;
y25.LO = 0; y25.UP = 100;
y31.LO = 0; y31.UP = 100;
y32.LO = 0; y32.UP = 200;
y33.LO = 0; y33.UP = 100;
y34.LO = 0; y34.UP = 100;
y35.LO = 0; y35.UP = 100;
z31.LO = 0; z31.UP = 100;
z32.LO = 0; z32.UP = 200;
z33.LO = 0; z33.UP = 100;
z34.LO = 0; z34.UP = 100;
z35.LO = 0; z35.UP = 100;

MODEL Pool2 /ALL/;

$onecho >bench.opt
  solvers conopt knitro minos
$offecho
pool2.optfile=1;
option nlp=bench;

SOLVE Pool2 USING NLP MAXIMIZING objval;
* End Pool2
```

Fig. 18.2. Expresia GAMS a aplicației G18 (Pool2).

Performanțele algoritmilor CONOPT, KNITRO şi MINOS sunt expuse în tabelul 18.1.

Tabelul 18.1

$n = 33$ (variabile), $m = 20$ (restricții)

	#iter	**time**	**vfo**
CONOPT	12	0.082	3500
KNITRO	124	0.450	1500
MINOS	6	0.090	1900

Soluția preferată este cea furnizată de CONOPT pentru care valoarea funcției obiectiv este 3500 unități. Observăm discrepanțele foarte mari între soluțiile optime locale furnizate de diferiți algoritmi de optimizare. Algoritmul CONOPT este singurul care poate identifica optimul problemei.

Este instructiv să menționăm că aplicația A40 (capitolul 5) din [Andrei, 2003, pg. 404] este o altă reprezentare a acestei probleme de optimizare, de data aceasta într-o formă abstractă. Algoritmul SPENBAR furnizează o altă soluție cu aceeaşi valoare a funcției obiectiv ca cea dată de CONOPT.

Problemele de amestec (de tip pooling şi blending) sunt probleme de optimizare biliniare şi neconvexe. Chiar dacă acestea sunt uşor de formulat, ele sunt considerate probleme greu de rezolvat. Comportarea lor în raport cu diferiți algoritmi de optimizare, aşa după cum s-a văzut în această aplicație, este tipică. Adică diferiți algoritmi de optimizare furnizează soluții optime locale foarte diferite.

Aplicația G19 (*HeatEx1*)

Proiectarea unei rețele de trei schimbătoare de căldură în serie care încălzeşte un flux de apă rece.

Un flux de apă rece trebuie încălzit de la 100^0 la 500^0 utilizând o rețea de trei schimbătoare de căldură cu diferite temperaturi de intrare, ca în figura 19.1. Pentru fiecare schimbător de căldură se cunoaşte temperatura apei la intrarea acestuia, precum şi alte caracteristici termice [Floudas, *et al.* 1999].

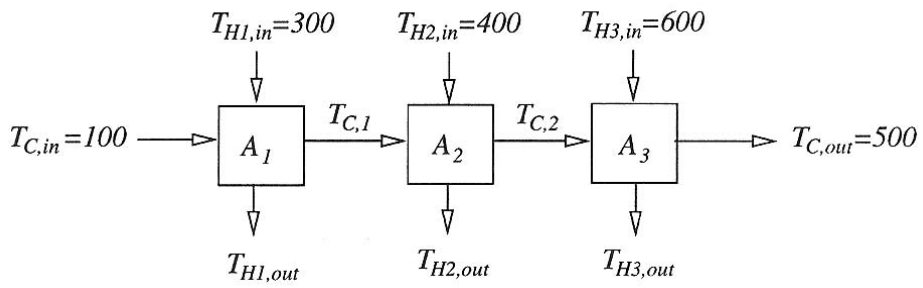

Fig. 19.1. O rețea de trei schimbătoare de căldură în serie.

Temperatura apei calde de la intrarea în cele trei schimbătoare de căldură: $T_{Hj,in}$, $j = 1,2,3,$ este cunoscută. Scopul este de a minimiza aria totală a celor trei schimbătoare de căldură. Din figura 19.1 putem scrie modelul matematic [Avriel şi Williams, 1971]:

$$\min(A_1 + A_2 + A_3)$$

referitor la:

$$T_{C,1} + T_{H1,out} - T_{C,in} - T_{H1,in} \le 0,$$
$$-T_{C,1} + T_{C,2} + T_{H2,out} - T_{H1,in} \le 0,$$
$$T_{H3,out} - T_{C,2} - T_{H3,in} + T_{C,out} \le 0,$$

$$A_1 - A_1 T_{H1,out} + \frac{FC_p}{U_1} T_{C,1} - \frac{FC_p}{U_1} T_{C,in} \le 0,$$

$$A_2 T_{C,1} - A_2 T_{H2,out} - \frac{FC_p}{U_2} T_{C,1} + \frac{FC_p}{U_2} T_{C,2} \le 0,$$

$$A_3 T_{C,2} - A_3 T_{H3,out} - \frac{FC_p}{U_3} T_{C,2} + \frac{FC_p}{U_3} T_{C,out} \le 0.$$

Marginile asupra variabilelor sunt:

$$100 \le A_1 \le 10000, \qquad 1000 \le A_2, A_3 \le 10000,$$
$$10 \le T_{C,1} \le 1000, \qquad 10 \le T_{C,2} \le 1000,$$
$$10 \le T_{H1,out} \le 1000,$$
$$10 \le T_{H2,out} \le 1000,$$
$$10 \le T_{H3,out} \le 1000.$$

Se consideră: $FC_p = 10^5$ şi $U = (120, 80, 40)$.

Cu acestea, expresia GAMS a acestei aplicații este arătată în figura 19.2.

```
$ONTEXT
Design of a heat exchanger network.
One cold stream must be heated from 100 F to 500 F using three hot
streams with different inlet temperatures. The goal is to minimize the
overall heat exchange area.
$OFFTEXT

SCALAR Tcin  inlet temperature of cold stream   /100/;
SCALAR Tcout outlet temperature of cold stream  /500/;
SCALAR Th1in inlet temperature of hot stream 1  /300/;
SCALAR Th2in inlet temperature of hot stream 2  /400/;
SCALAR Th3in inlet temperature of hot stream 3  /600/;
SCALAR FCp   flowrate-heat capacity of cold stream /100000/;
SCALAR U1    heat transfer coefficient for first heat exchanger /120/;
SCALAR U2    heat transfer coefficient for second heat exchanger /80/;
SCALAR U3    heat transfer coefficient for third heat exchanger  /40/;

VARIABLES
         A1  area of first heat exchanger
         A2  area of second heat exchanger
         A3  area of third heat exchanger
         Tc1 temperature of cold stream after first heat exchanger
         Tc2 temperature of cold stream after second heat exchanger
         Th1out  outlet temperature of stream 1
         Th2out  outlet temperature of stream 2
         Th3out  outlet temperature of stream 3
         objval  objective function variable;

FREE VARIABLES    objval;


EQUATIONS
         f Objective function
```

```
         f1
         f2
         f3
         f4
         f5
         f6  ;

f  .. objval =e=A1 + A2 + A3;
f1  .. Tc1 + Th1out - Tcin - Th1in =l= 0;
f2  .. -Tc1 + Tc2 + Th2out - Th1in =l= 0;
f3  .. Th3out - Tc2 - Th3in + Tcout =l= 0;
f4  .. A1 - A1*Th1out + FCp/U1*Tc1 - FCp/U1*Tcin =l= 0;
f5  .. A2*Tc1 - A2*Th2out - FCp/U2*Tc1 + FCp/U2*Tc2 =l= 0;
f6  .. A3*Tc2 - A3*th3out - FCp/U3*Tc2 + FCp/U3*Tcout =l= 0;

* Bounds
A1.LO = 100;     A1.UP = 10000;
A2.LO = 1000;    A2.UP = 10000;
A3.LO = 1000;    A3.UP = 10000;
Tc1.LO = 10;     Tc1.UP = 1000;
Tc2.LO = 10;     Tc2.UP = 1000;
Th1out.LO = 10;  Th1out.UP = 1000;
Th2out.LO = 10;  Th2out.UP = 1000;
Th3out.LO = 10;  Th3out.UP = 1000;

MODEL HeatEx1 /ALL/;

$onecho >bench.opt
  solvers conopt knitro minos
$offecho
HeatEx1.optfile=1;
option nlp=bench;

SOLVE HeatEx1 USING NLP MINIMIZING objval;
* End HeatEx1
```

Fig. 19.2. Expresia GAMS a aplicației G19 (HeatEx1).

Soluția aplicației este următoarea:

$$A_1 = 1026.948, \quad A_2 = 1000.0, \quad A_3 = 5485.282,$$

$$T_{C,1} = 265.060, \quad T_{C,2} = 280.589,$$

$$T_{H1,out} = 134.940, \quad T_{H2,out} = 284.471, \quad T_{H3,out} = 380.589.$$

Tabelul 19.1 arată performanțele algoritmilor utilizați pentru rezolvarea acestei probleme.

Tabelul 19.1

$n = 9$ (variabile), $m = 7$ (restricții)

	#iter	time	vfo
CONOPT	16	0.039	7512.230
KNITRO	20	0.100	7512.230
MINOS	69	0.273	7512.230

O variantă (mai simplă) a acestei probleme, ca o formulare alternativă, este aplicația A31 (capitolul 5) din [Andrei, 2003, pg. 392], în care problema este prezentată sub forma abstractă [Dembo, 1976], [Floudas, *et al.*, 1999, p.90].

Aplicația G20 (*HeatEx2*)
Proiectarea unei rețele de trei schimbătoare de căldură în paralel, cu recirculație, care încălzeşte un flux de apă rece.
Este vorba de un flux de apă rece care trebuie încălzită de la 100^0 la 200^0 utilizând trei schimbătoare de căldură plasate în paralel, ca în figura 20.1 .

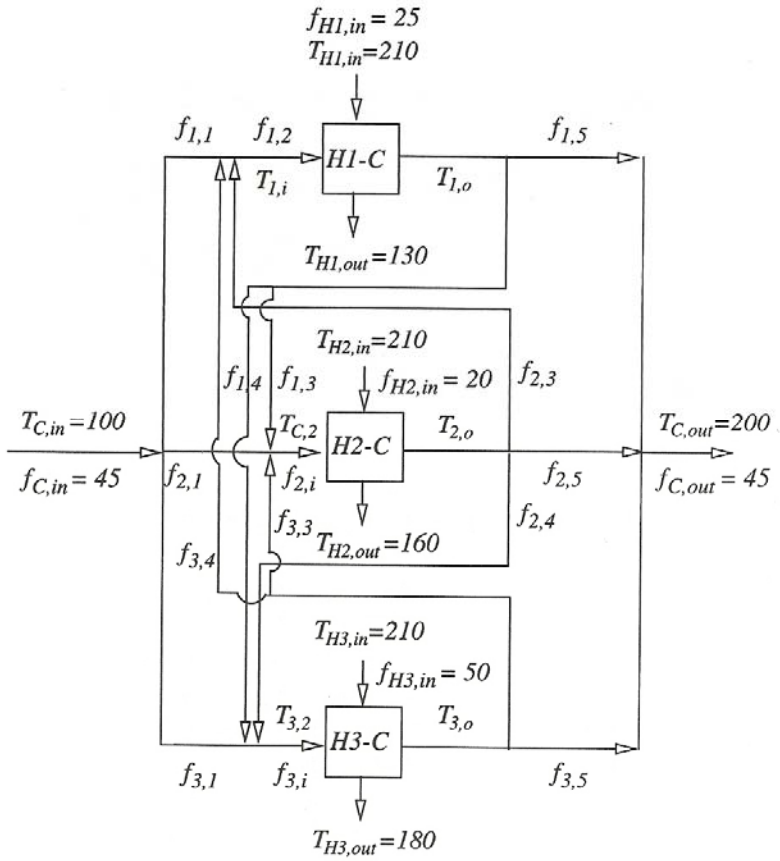

Fig. 20.1. O rețea de trei schimbătoare de căldură în paralel.

Din figura 20.1 vedem că cele trei schimbătoare de căldură sunt plasate în paralel cu recirculție totală, adică debitul de apă încălzită de la ieşirea fiecărui schmbător este reintrodus în celelalte. Pentru fiecare schimbător de căldură se cunoaşte temperatura de intrare a apei calde $T_{Hj.in}$, a celei de ieşire $T_{Hj,out}$, precum şi debitul de intrare de apă caldă $f_{Hj,in}$, $j = 1,2,3$. Variabilele problemei sunt $T_{j,i}$, $T_{j,o}$ ca temperatura la intrarea şi respectiv ieşirea schimbătorului j, $j = 1,2,3$. $f_{k,j}$ debitul de apă încălzită care intră, iese sau care este recirculată în schimbătorul k, $k = 1,2,3$. $\Delta T_{j,1}$ diferența de temperatură pe apă caldă pentru schimbătorul j, şi $\Delta T_{j,2}$ diferența de temperatură pe apă rece pentru schimbătorul j, $j = 1,2,3$.

Modelul matematic al acestei rețele de schimbătoare de căldură, descris în [Floudas, *et al.* 1999], [Floudas şi Ciric, 1988], este următorul:

$$\min 1300\left(\frac{2000}{(\Delta T_{11}\Delta T_{12})/3+(\Delta T_{11}+\Delta T_{12})/6}\right)^{0.6}+$$
$$1300\left(\frac{1000}{2(\Delta T_{21}\Delta T_{22})/3+(\Delta T_{21}+\Delta T_{22})/6}\right)^{0.6}+$$
$$1300\left(\frac{1500}{4(\Delta T_{31}\Delta T_{32})/3+(\Delta T_{31}+\Delta T_{32})/6}\right)^{0.6},$$

referitor la:

$$f_{1,1}+f_{2,1}+f_{3,1}=45,$$
$$f_{1,1}+f_{2,3}+f_{3,4}-f_{1,2}=0,$$
$$f_{2,1}+f_{1,3}+f_{3,3}-f_{2,2}=0,$$
$$f_{3,1}+f_{1,4}+f_{2,4}-f_{3,2}=0,$$
$$f_{1,5}+f_{1,3}+f_{1,4}-f_{1,2}=0,$$
$$f_{2,5}+f_{2,3}+f_{2,4}-f_{2,2}=0,$$
$$f_{3,5}+f_{3,3}+f_{3,4}-f_{3,2}=0,$$
$$100f_{1,1}+T_{2,o}f_{2,3}+T_{3,o}f_{3,4}-T_{1,i}f_{1,2}=0,$$
$$100f_{2,1}+T_{1,o}f_{1,3}+T_{3,o}f_{3,3}-T_{2,i}f_{2,2}=0,$$
$$100f_{3,1}+T_{1,o}f_{1,4}+T_{2,o}f_{2,4}-T_{3,i}f_{3,2}=0,$$
$$f_{1,2}(T_{1,0}-T_{1,i})=2000,$$
$$f_{2,2}(T_{2,0}-T_{2,i})=1000,$$
$$f_{3,2}(T_{3,0}-T_{3,i})=1500,$$
$$\Delta T_{11}+T_{1,o}=210,$$
$$\Delta T_{12}+T_{1,i}=130,$$
$$\Delta T_{21}+T_{2,o}=210,$$
$$\Delta T_{22}+T_{2,i}=160,$$
$$\Delta T_{31}+T_{3,o}=210,$$
$$\Delta T_{32}+T_{3,i}=180.$$

Marginile asupra variabilelor sunt următoarele:

$$10\le\Delta T_{1,1}\le 110,\quad 10\le\Delta T_{1,2}\le 110,$$
$$10\le\Delta T_{2,1}\le 110,\quad 10\le\Delta T_{2,2}\le 110,$$
$$10\le\Delta T_{3,1}\le 110,\quad 10\le\Delta T_{3,1}\le 110,$$
$$0\le f_{1,j}\le 45,\quad j=1,\ldots,5,$$

$$0 \le f_{2,j} \le 45, \quad j = 1,\ldots,5,$$
$$0 \le f_{3,j} \le 45, \quad j = 1,\ldots,5,$$
$$100 \le T_{j,i} \le 200, \quad j = 1,\ldots,3,$$
$$100 \le T_{j,o} \le 200, \quad j = 1,\ldots,3,$$

Problema de optimizare are funcția obiectiv convexă şi restricțiile sunt liniare şi biliniare. Expresia GAMS a acestei aplicații este prezentată în figura 20.2.

```
$ONTEXT
The problem is the heat exchanger network design and involves three hot
streams and one cold stream.
$OFFTEXT

VARIABLES
         dT11  temperature difference at hot end of exchanger H1-C
         dT12  temperature difference at cold end of exchanger H1-C
         dT21  temperature difference at hot end of exchanger H2-C
         dT22  temperature difference at cold end of exchanger H2-C
         dT31  temperature difference at hot end of exchanger H3-C
         dT32  temperature difference at cold end of exchanger H3-C
         f11
         f12, f13, f14, f15, f21, f22, f23, f24, f25, f31, f32
         f33, f34, f35, t1i, t1o, t2i, t2o, t3i, t3o
      objval   Automatically generated objective function variable;

FREE VARIABLES    objval;

EQUATIONS
         g1,  g2,  g3,  g4,  g5,  g6, g7, g8, g9, g10, g11, g12, g13
         g14, g15, g16, g17, g18, g19
         f Objective function;

f  .. objval =e=1300*(2000/(1/3*dT11*dT12+1/6*(dT11+dT12)))**0.6+
                1300*(1000/(2/3*dT21*dT22+1/6*(dT21+dT22)))**0.6+
                1300*(1500/(2/3*dT31*dT32+1/6*(dT31+dT32)))**0.6;
g1  .. f11+f21+f31 =e= 45;
g2  .. f11+f23+f34-f12 =e= 0;
g3  .. f21+f13+f33-f22 =e= 0;
g4  .. f31+f14+f24-f32 =e= 0;
g5  .. f15+f13+f14-f12 =e= 0;
g6  .. f25+f23+f24-f22 =e= 0;
g7  .. f35+f33+f34-f32 =e= 0;
g8  .. 100*f11+t2o*f23+t3o*f34-t1i*f12 =e= 0;
g9  .. 100*f21+t1o*f13+t3o*f33-t2i*f22 =e= 0;
g10  .. 100*f31+t1o*f14+t2o*f24-t3i*f32 =e= 0;
g11  .. f12*t1o-f12*t1i =e= 2000;
g12  .. f22*t2o-f22*t2i =e= 1000;
g13  .. f32*t3o-f32*t3i =e= 1500;
g14  .. dT11+t1o =e= 210;
g15  .. dT12+t1i =e= 130;
g16  .. dT21+t2o =e= 210;
g17  .. dT22+t2i =e= 160;
g18  .. dT31+t3o =e= 210;
g19  .. dT32+t3i =e= 180;

dT11.LO = 10; dT11.UP = 110;
dT12.LO = 10; dT12.UP = 110;
dT21.LO = 10; dT21.UP = 110;
dT22.LO = 10; dT22.UP = 110;
```

```
dT31.LO = 10; dT31.UP = 110;
dT32.LO = 10; dT32.UP = 110;
f11.LO = 0; f11.UP = 45;
f12.LO = 0; f12.UP = 45;
f13.LO = 0; f13.UP = 45;
f14.LO = 0; f14.UP = 45;
f15.LO = 0; f15.UP = 45;
f21.LO = 0; f21.UP = 45;
f22.LO = 0; f22.UP = 45;
f23.LO = 0; f23.UP = 45;
f24.LO = 0; f24.UP = 45;
f25.LO = 0; f25.UP = 45;
f31.LO = 0; f31.UP = 45;
f32.LO = 0; f32.UP = 45;
f33.LO = 0; f33.UP = 45;
f34.LO = 0; f34.UP = 45;
f35.LO = 0; f35.UP = 45;
t1i.LO = 100; t1i.UP = 200;
t1o.LO = 100; t1o.UP = 200;
t2i.LO = 100; t2i.UP = 200;
t2o.LO = 100; t2o.UP = 200;
t3i.LO = 100; t3i.UP = 200;
t3o.LO = 100; t3o.UP = 200;

MODEL HeatEx2 /ALL/;
$onecho >bench.opt
  solvers conopt knitro minos
$offecho
HeatEx2.optfile=1;
option nlp=bench;
SOLVE HeatEx2 USING NLP MINIMIZING objval;
```

Fig. 20.2. Expresia GAMS a aplicației G20 (HeatEx2).

Soluția aplicației este următoarea:

$$\Delta T = \begin{bmatrix} 52.857 & 30.000 \\ 10.000 & 60.000 \\ 10.000 & 22.857 \end{bmatrix}, \qquad f = \begin{bmatrix} 35 & 35 & 0 & 35 & 0 \\ 10 & 10 & 0 & 0 & 10 \\ 0 & 35 & 0 & 0 & 35 \end{bmatrix},$$

$$T_{1,i} = 100, \qquad T_{1,o} = 157.143,$$

$$T_{2,i} = 100, \qquad T_{2,o} = 200,$$

$$T_{3,i} = 157.143, \quad T_{3,o} = 200.$$

Performanțele algoritmilor sunt date în tabelul 20.1.

Tabelul 20.1

$n = 28$ (variabile), $m = 20$ (restricții)

	#iter	time	Vfo
CONOPT	16	0.016	10077.7754
KNITRO	168	0.380	10077.7754
MINOS	13	0.188	10077.78

O altă versiune privind proiectarea configurației optime a unei rețele de trei schimbătoare de căldură este prezentată în aplicația A33 (capitolul 5) din [Andrei, 2003, pg. 395].

Aplicația G21 (*DifuzieT*)
Calculul difuziei temperaturii într-o placă dreptunghiulară, cu conductivitate termică eterogenă, simetric distribuită, cu condiții pe frontieră şi cu diverse surse de căldură plasate în interiorul ei.
Prima lege a termodinamicii are forma:

$$Q = W + \frac{\mathrm{d}U}{\mathrm{d}t}, \tag{21.1}$$

unde Q rata de transfer a căldurii, W este lucrul mecanic şi U este energia termică internă a sistemului, aşa după cum se vede în figura 21.1. $\mathrm{d}U/\mathrm{d}t$ este rata de schimbare în timp a energiei termice interne.
În general, analiza procesului de transfer a căldurii se poate face fără a ne referi la lucrul mecanic făcut de sistem. Totuşi, în analiza sistemelor reale, transferul de căldură trebuie combinat cu lucrul mecanic.

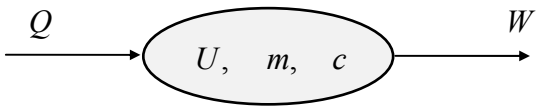

Fig. 21.1. Prima lege a termodinamicii pentru un sistem închis.

Când sistemul nu transferă lucru mecanic, atunci prima lege a termodinamicii este:

$$Q = \frac{\mathrm{d}U}{\mathrm{d}t} = mc\frac{\mathrm{d}T}{\mathrm{d}t}, \tag{21.2}$$

unde m este masa sistemului, c este căldura specifică şi T este temperatura sistemului. Ecuația de transfer a căldurii se scrie conform *legii lui Fourier*. *Fluxul de căldură q rezultat din conducția termică este proporțional cu mărimea gradientului temperaturii cu semn schimbat:*

$$q = -V\nabla T,$$

unde V este conductivitatea termică.
Pe componente, legea lui Fourier este:

$$q_x = -V\frac{\mathrm{d}T}{\mathrm{d}x}, \quad q_y = -V\frac{\mathrm{d}T}{\mathrm{d}y}, \quad q_z = -V\frac{\mathrm{d}T}{\mathrm{d}z}.$$

În general, conductivitatea termică V este o funcție de punct şi temperatură. Totuşi, majoritatea materialelor sunt omogene şi deci putem considera $V = V(T)$.

Cu acestea, să considerăm un volum R mărginit de o suprafață S şi un mic element diferențial $\mathrm{d}S$, centrat într-un punct de pe suprafața S. Referindu-ne la elementul diferențial $\mathrm{d}S$, se pot asocia doi vectori: primul este vectorul n normal la suprafața S, şi al doilea $q = -V\nabla T$ fluxul de căldură prin suprafața S. În acelaşi timp putem considera că în volumul R este distribuită o sursă

(volumetrică) de căldură f. Aceasta poate rezulta, de exemplu, din anumite reacții chimice, nucleare sau electrice. Ca atare, căldura care *iese* prin elementul de suprafață $\mathrm{d}S$ este:

$$(-V\nabla T).(n\mathrm{d}S). \tag{21.3}$$

Considerând acum şi căldura generată (sau consumată) în volumul R, atunci căldura totală care curge *prin* S în volumul R este:

$$Q = -\int_S (-V\nabla T).(ndS) + \int_R fdR. \tag{21.4}$$

Dar, rata de creştere a energiei interne în volumul R este:

$$\frac{\mathrm{d}U}{\mathrm{d}t} = \int_R \left(\rho c \frac{\partial T}{\partial t}\right)\mathrm{d}R, \tag{21.5}$$

unde ρ este densitatea substanței din volumul R. Cu acestea, din (21.2) obținem imediat:

$$\int_S V\nabla T.n\mathrm{d}S = \int_R \left[\rho c \frac{\partial T}{\partial t} - f\right]\mathrm{d}R. \tag{21.6}$$

Aplicând formula Gauss-Ostrogradski, obținem:

$$\int_R \left(\nabla.V\nabla T - \rho c \frac{\partial T}{\partial t} + f\right)dR = 0. \tag{21.7}$$

Deoarece volumul R este arbitrar, rezultă:

$$\rho c \frac{\partial T}{\partial t} = \nabla.V\nabla T + f, \tag{21.8}$$

cunoscută ca *ecuația de transport* (*difuzie*) *a căldurii*.

Având în vedere discuțiile de mai sus, această ecuație descrie transportul căldurii în medii incompresibile (nu am inclus lucrul mecanic efectuat de sistem) şi fără convecție (mediul nu execută nici o mişcare relativă).

Pentru rezolvarea acestei ecuații, adică pentru determinarea transportului de căldură în medii trebuie precizate condițiile în care se execută aceste transport. Aceasta revine la a preciza informații asupra conductivității materialului prin care se face transportul de căldură, a densității şi căldurii specifice a acestuia, precum şi a condițiilor pe frontiera domeniului în care se înregistrează difuzia de căldură. Cu acestea rezolvarea ecuației căldurii se face prin discretizarea domeniului şi deci calculul temperaturii în punctele rețelei de discretizare. Se pot considera mai multe situații pe care le vom considera în secțiunile următoare.

Să considerăm acum calculul temperaturii staționare într-o placă dreptunghiulară [Andrei, 2004a]. În cazul staționar caracteristicile problemei nu se schimbă în timp. Vom considera o placă dreptunghiulară pentru care conductivitatea termică V [$Wm^{-1}K^{-1}$] este eterogenă, simetric distribuită de-a lungul plăcii. Atunci ecuația de transport a căldurii devine:

$$\nabla.V\nabla T + f = 0. \tag{21.9}$$

Pentru rezolvarea ecuației (21.9) vom aplica operatorul diferențial ∇ atât conductivității termice cât şi gradientului temperaturii obținând următoarea expresie a ecuației căldurii:

$$V\nabla.\nabla T + \nabla T.\nabla V + f = 0. \qquad (21.10)$$

Utilizând aproximațiile prin diferențe finite, în două dimensiuni, obținem sistemul:

$$V_{i,j}\frac{(T_{i+1,j} - 2T_{i,j} + T_{i-1,j})}{(\Delta x)^2} + V_{i,j}\frac{(T_{i,j+1} - 2T_{i,j} + T_{i,j-1})}{(\Delta y)^2}$$
$$+\frac{(V_{i+1,j} - V_{i-1,j})}{2(\Delta x)}\frac{(T_{i+1,j} - T_{i-1,j})}{2(\Delta x)} + \frac{(V_{i,j+1} - V_{i,j-1})}{2(\Delta y)}\frac{(T_{i,j+1} - T_{i,j-1})}{2(\Delta y)} + f_{i,j} = 0,$$

pentru $i = 1,\ldots,n_x$ şi $j = 1,\ldots,n_y$, unde n_x şi respectiv n_y sunt numărul punctelor de discretizare de-a lungul laturilor plăcii [McKinney şi Savitsky, 2003].

Pentru determinarea distribuției temperaturii în nodurile acestei rețele trebuie să rezolvăm acest sistem de ecuații algebrice la care se adaugă condițiile pe frontiera plăcii. Considerăm cazul în care pe frontiera plăcii se menține o temperatură constantă nulă cu excepția celulelor (I1,J5), (I1,J6), (I5,J1) şi (I6,J1) în care temperatura are 100^0 C. Dacă $n_x = 20$ şi $n_y = 20$, atunci reprezentarea în GAMS a calculului temperaturii staționare este cea din figura 21.2.

```
$ontext
Stationary temperature field in a rectangular area.
$offtext

   set x /i1*i20/
   set y /j1*j20/;

   set inside(x,y);
       inside(x,y)                  = yes;
       inside(x,y)$(ord(x)=1)       = no;
       inside(x,y)$(ord(x)=card(x)) = no;
       inside(x,y)$(ord(y)=1)       = no;
       inside(x,y)$(ord(y)=card(y)) = no;

* Temperature supply

   parameter supply(x,y);
             supply(x,y)          := 0;
             supply('i17','j17')  := 25000;
             supply('i5','j5')    := 10000;

* Parameters determination

   scalar dx /0.1/;
   scalar dy /0.1/;
   scalar c /1./;
   scalar ro /1./;

   TABLE v(x,y)

     j1  j2  j3  j4  j5  j6  j7  j8  j9 j10 j11 j12 j13 j14 j15 j16 j17 j18 j19 j20
i1  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5
i2  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5
i3  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5
i4  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5
i5  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 1.0 1.0 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5
i6  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 1.0 1.0 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5
i7  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 1.0 1.0 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5
i8  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 1.0 1.0 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5
```

```
i9  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 1.0 1.0 1.0 1.0 0.5 0.5 0.5 0.5 0.5 0.5
i10 0.5 0.5 0.5 0.5 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 0.5 0.5 0.5 0.5 0.5 0.5
i11 0.5 0.5 0.5 0.5 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 0.5 0.5 0.5 0.5 0.5 0.5
i12 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 1.0 1.0 1.0 1.0 1.0 1.0 0.5 0.5 0.5 0.5 0.5
i13 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 1.0 1.0 1.0 1.0 1.0 1.0 1.0 0.5 0.5 0.5 0.5
i14 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 1.0 1.0 1.0 1.0 1.0 0.5 0.5 0.5
i15 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 1.0 1.0 1.0 1.0 1.0 0.5 0.5
i16 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 1.0 1.0 1.0 1.0 0.5 0.5
i17 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 1.0 1.0 1.0 0.5 0.5
i18 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5
i19 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5
i20 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5

   variables
             obj
             t(x,y);

   t.lo(x,y)  = 0;
   t.up(x,y)  =  200;

   t.l(x,y) = 0;

* Boundary conditions

   t.fx(x,y)$(ord(x)=1)       = 0.0;
   t.fx(x,y)$(ord(x)=card(x)) = 0.0;
   t.fx(x,y)$(ord(y)=1)       = 0.0;
   t.fx(x,y)$(ord(y)=card(y)) = 0.0;

   t.fx('i1','j10') = 100;
   t.fx('i1','j11') = 100;
   t.fx('i10','j1') = 100;
   t.fx('i11','j1') = 100;

* Equations of the model

   equation
         temp(x,y)
         ben;
   temp(x,y)$(inside(x,y)).. supply(x,y)/c/ro
                 + v(x,y)*(t(x-1,y) - 2*t(x,y) + t(x+1,y))/dx/dx
                 + v(x,y)*(t(x,y-1) - 2*t(x,y) + t(x,y+1))/dy/dy
                 +(v(x,y+1) - v(x,y-1))*(t(x,y+1)-t(x,y-1))/dy/dy/2.0
                 +(v(x+1,y) - v(x-1,y))*(t(x+1,y)-t(x-1,y))/dx/dx/2.0
                 =e= 0;

*   ben.. obj =e= sum(x,t(x,'j2'));
    ben.. obj =e= 0;

   model DifuzieT /all/;

$onecho >bench.opt
  solvers conopt knitro minos
$offecho
DifuzieT.optfile=1;
option nlp=bench;

solve DifuzieT using nlp minimizing obj;

* Put the solution

  file res1 /difuziet.txt/
  file res2 /difuziet.dat/
  put res1;
  loop(x, put x.tl:6; loop(y, put t.l(x,y):6:2; ); put /;) put /;
          put " inside 20 " /;
```

```
  loop(x, put x.tl:6; loop(y, put inside(x,y):6; )put /;);
          put "inside 20 " /;
  loop(x, put x.tl:6; loop(y, put supply(x,y):6; ); put /;);
  put res2;
  loop(x, loop(y, put t.l(x,y):4:0; ); put /;); put /;
* End DifuzieT
***************************************************** March 18, 2004
```

Fig. 21.2. Expresia GAMS a aplicației G21 (DifuzieT).

În figura 21.3 se arată suprafața temperaturii care se realizează în placă în care se identifică imediat cele două surse de căldură plasate pe frontiera plăcii, precum şi curbele de temperatură constantă din placă şi modul de distribuție a căldurii în placă.

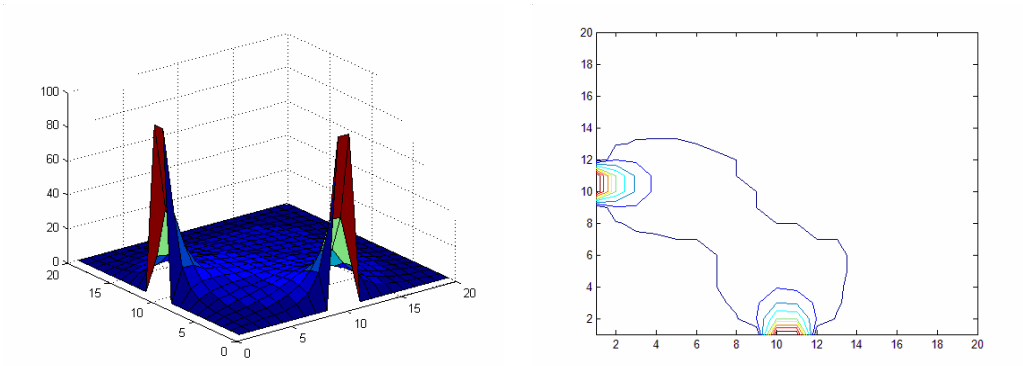

Fig. 21.3. Suprafața temperaturii şi curbele de temperatură constantă din placă, două surse calde.

Considerând acum situația în care asupra plăcii se aplică suplimentar o sursă de căldură plasată în centrul celulei (I5,J5) de valoare 10000, atunci distribuția temperaturii de-a lungul plăcii şi curbele de temperatură constantă sunt cele din figura 21.4.

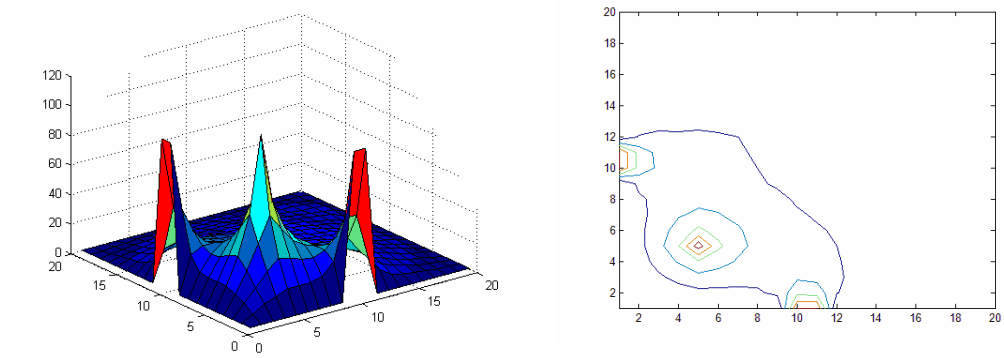

Fig. 21.4. Suprafața temperaturii şi curbele de temperatură constantă din placă, trei surse calde.

Se observă că simetria distribuției conductivității termice în placă şi a condițiilor pe frontieră determină simetria distribuției temperaturii în placă. Pentru obținerea soluției problemei de difuzie a căldurii în modelul GAMS din figura 21.2 se introduce instrucțiunea `SUPPLY('I5','J5'):=10000;` care precizează locul şi valoarea sursei suplimentare de căldură.

În cele ce urmează să considerăm cazul în care suplimentar se introduce o altă sursă de căldură în celula (I17,J17) cu o valoare de 25000. Figura 21.5 arată distribuția temperaturii de-a lungul plăcii şi curbele de temperatură constantă.

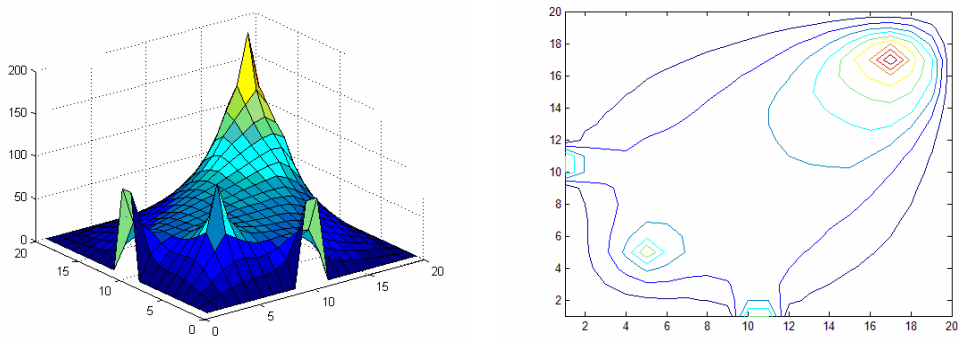

Fig. 21.5. Suprafața temperaturii şi curbele de temperatură constantă din placă, patru surse calde.

Performanțele algoritmilor sunt prezentate în tabelul 21.1.

Tabelul 21.1

$n = 397$ (variabile), $m = 325$ (restricții)

	#iter	time	vfo
CONOPT	5	0.041	0
KNITRO	5	0.250	0
MINOS	1 / 48†	0.061	0

† 48 iterații minore.

Aplicația G22 (*Flow*)

Curgerea staționară a unui fluid incompresibil intr-o arie dreptunghiulară.

Ecuațiile care descriu curgerea sunt cele de conservare a momentului liniar a unui fluid incompresibil în planul orizontal şi ecuația de conservare a masei (ecuația de continuitate). Dacă $v = v(x,t)$ şi $A = A(x,t)$ sunt viteza fluidului şi respectiv aria secțiunii transversale a canalului prin care curge fluidul din punctul x de la momentul t, atunci *principiul conservării masei* spune că *variația masei de fluid din intervalul* $[a,b]$ *este egală cu fluxurile prin punctele* a *şi* b *plus sursele de fluid din acest interval*. Deci:

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_a^b \rho A \mathrm{d}x = \rho v A\big|_a - \rho v A\big|_b + \int_a^b \rho q \mathrm{d}x,$$

unde ρ este densitatea fluidului şi $q = q(x,t)$ este pierderea sau câştigul de fluid prin pereții laterali ai canalului. Deoarece intervalul $[a,b]$ este arbitrar, rezultă ecuația de conservare a masei:

$$\frac{\mathrm{d}A}{\mathrm{d}t} = -\frac{\mathrm{d}}{\mathrm{d}x}(vA) + q\,. \tag{22.1}$$

Principiul de conservare a momentului zice că *variația momentului în intervalul* $[a,b]$ *este egală cu suma forțelor care acționează asupra fluidului din acelaşi interval*. Suplimentar trebuie să se considere şi fluxurile prin punctele a şi b plus sursele de fluid din acest interval. Deci:

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_a^b \rho vA\mathrm{d}x = \rho v^2 A\Big|_a - \rho v^2 A\Big|_b + \int_a^b \rho\mu q\mathrm{d}x - \int_a^a \rho Ag\left(h_x + S_f\right)\mathrm{d}x\,,$$

unde μ este viteza de curgere a fluidului prin pereții canalului, g este accelerația gravitațională, $h = h(x,t)$ este înălțimea apei față de nivelul mării, $S_f(v^2, y)$ este coeficientul de frecare, y este nivelul fluidului din canal față de fundul acestuia. Forțele care acționează asupra fluidului sunt frecarea, gravitația şi cele datorate variației presiunii. Deoarece intervalul $[a,b]$ este arbitrar, rezultă ecuația de conservare a momentului:

$$\frac{\mathrm{d}}{\mathrm{d}t}(vA) = -\frac{\mathrm{d}}{\mathrm{d}t}(v^2 A) + q\mu - Ag\left(h_x + S_f\right)\,. \tag{22.2}$$

Ecuațiile (22.1), (22.2) sunt *ecuațiile Saint-Venant de curgere a fluidelor incompresibile* [McKinney, Savitsky, 2003].

Pentru conservarea momentului, în direcția x avem:

$$\frac{\partial V_x}{\partial t} + V_x\frac{\partial V_x}{\partial x} + V_y\frac{\partial V_y}{\partial y} = -\frac{1}{\rho}\frac{\partial P}{\partial x} + \mu\nabla^2 V_x\,, \tag{22.3}$$

în direcția y:

$$\frac{\partial V_y}{\partial t} + V_y\frac{\partial V_y}{\partial x} + V_y\frac{\partial V_y}{\partial y} = -\frac{1}{\rho}\frac{\partial P}{\partial y} + \mu\nabla^2 V_y\,. \tag{22.4}$$

Ecuația de continuitate se scrie ca

$$\frac{\partial V_x}{\partial x} + \frac{\partial V_y}{\partial y} = 0, \tag{22.5}$$

unde: V_x este componenta în direcția x a vitezei [m/sec], V_y este componenta în direcția y a vitezei [m/sec], P presiunea [pa], μ viscozitatea cinematică [m2/sec] şi ρ densitatea fluidului [Kg/m3].

Observăm că acest model de curgere se aplică în cazul în care viscozitatea este constantă şi greutatea fluidului acționează perpendicular pe planul $x - y$. Ca atare, forța gravitațională este absentă din ecuații.

Cu acestea să considerăm curgerea unui fluid incompresibil într-o arie dreptunghiulară în care într-o latură există un aport de fluid care iese prin latura

opusă. Pe latura de intrare a fluidului condițiile la limită sunt: $V_x = 0.5$ [m/sec] şi $V_y = 0$ [m/sec]. Pe latura opusă a ariei dreptunghiulare condițiile sunt de tipul: $\frac{\partial V_x}{\partial x} = 0$ şi $V_y = 0$. Pe celelalte laturi ale ariei dreptunghiulare $V_x = 0$ [m/sec] şi $V_y = 0$ [m/sec].

Pentru calculul curgerii fluidului vom proceda la o discretizare a domeniului considerat. Ecuațiile de conservare a momentului nu ridică probleme, totuşi ecuația de continuitate introduce anumite dificultăți deoarece aceasta nu este o ecuație evolutivă. Viteza fluidului este calculată pe segmentele de linie care uneşte punctele de discretizare. Pe liniile paralele cu axa x se calculează V_x, iar pe cele paralele cu axa y, V_y. Cu acestea ecuația de continuitate are următoarea expresie discretizată:

$$\frac{V_{x,i,j} - V_{x,i-1,j}}{\Delta x} + \frac{V_{y,i,j} - V_{y,i,j-1}}{\Delta y} = 0. \tag{22.6}$$

Presiunea se calculează în nodurile de discretizare, unde gradientul de presiune se calculează din viteză, adică

$$\left.\frac{\partial P}{\partial x}\right|_{i,j} = \frac{P_{i+1,j} - P_{i,j}}{\Delta x}, \tag{22.7}$$

$$\left.\frac{\partial P}{\partial y}\right|_{i,j} = \frac{P_{i,j+1} - P_{i,j}}{\Delta y}. \tag{22.8}$$

Termenii difuzivi din (22.3) şi (22.4) sunt calculați sub forma:

$$\mu\nabla^2 V_x \approx \mu\left(\frac{V_{x,i+1,j} - 2V_{x,i,j} + V_{x,i-1,j}}{(\Delta x)^2} + \frac{V_{x,i,j+1} - 2V_{x,i,j} + V_{x,i,j-1}}{(\Delta y)^2}\right), \tag{22.9}$$

$$\mu\nabla^2 V_y \approx \mu\left(\frac{V_{y,i+1,j} - 2V_{y,i,j} + V_{y,i-1,j}}{(\Delta x)^2} + \frac{V_{y,i,j+1} - 2V_{y,i,j} + V_{y,i,j-1}}{(\Delta y)^2}\right). \tag{22.10}$$

Rezolvarea sistemului de ecuații discretizate de mai sus nu conduce la o soluție, deoarece nu întotdeauna este posibil să aproximăm diferențialele prin diferențe finite. Şi aceasta deoarece erorile introduse de aproximațiile algebrice sunt prea mari. Dacă analizăm sistemul de ecuații (22.3) şi (22.4) vedem că aceste ecuații conțin componentele vitezei legate de gradientul de presiune ∇P. Dacă P nu este legat de condițiile pe frontieră, atunci toate erorile de aproximare vor părăsi domeniul considerat prin frontiera acestuia. În a treia ecuație (22.5) vectorul vitezei este legat de condițiile pe frontieră. Ca atare, în calculul ecuației de continuitate ne aşteptăm la erori în fiecare punct. Pentru a depăşi această dificultate, vom rescrie ecuația de continuitate într-o formă uşor modificată [McKinney şi Savitsky, 2003]:

$$\frac{\partial V_x}{\partial x}+\frac{\partial V_y}{\partial y}=\delta, \tag{22.11}$$

unde δ este eroarea în aproximarea diferențialelor care trebuie minimizată. Cu acestea sistemul final de ecuații care trebuie rezolvat este:

$$-\frac{1}{\rho}\frac{\partial P}{\partial x}+\mu\nabla^2 V_x=0, \tag{22.12}$$

$$-\frac{1}{\rho}\frac{\partial P}{\partial y}+\mu\nabla^2 V_y=0, \tag{22.13}$$

$$\frac{\partial V_x}{\partial x}+\frac{\partial V_y}{\partial y}=\delta. \tag{22.14}$$

Prin discretizare, sistemul (22.12)-(22.14), devine:

$$\frac{1}{\rho}\frac{P_{i+1,j}-P_{i,j}}{\Delta x}=\mu\left(\frac{V_{x,i+1,j}-2V_{x,i,j}+V_{x,i-1,j}}{(\Delta x)^2}+\frac{V_{x,i,j+1}-2V_{x,i,j}+V_{x,i,j-1}}{(\Delta y)^2}\right),$$

$$\frac{1}{\rho}\frac{P_{i,j+1}-P_{i,j}}{\Delta y}=\mu\left(\frac{V_{y,i+1,j}-2V_{y,i,j}+V_{y,i-1,j}}{(\Delta x)^2}+\frac{V_{y,i,j+1}-2V_{y,i,j}+V_{y,i,j-1}}{(\Delta y)^2}\right),$$

$$\frac{V_{x,i,j}-V_{x,i-1,j}}{\Delta x}+\frac{V_{y,i,j}-V_{y,i,j-1}}{\Delta y}=\delta_{i,j}.$$

Cu acestea să considerăm o discretizare a domeniului curgerii în care axa x este secționată în 15 intervale, iar axa y în 20, unde $\Delta x=\Delta y=1$. Considerăm densitatea fluidului egală cu 1000 [Kg/m3]. Viscozitatea fluidului este egală cu 0.005 [m2/sec]. Curgerea fluidului este paralelă cu axa x. Considerăm că în intervalul [u5, u15] de pe axa y avem condițiile la limită: $V_x=0.5$ [m/sec] şi $V_y=0$ [m/sec]. Pe latura opusă a domeniului condițiile la limită sunt: $\frac{\partial V_x}{\partial x}=0$ şi $V_y=0$. Pe toate celelalte intervale ale discretizării $V_x=0$ [m/sec] şi $V_y=0$ [m/sec].

Cu acestea expresia GAMS a acestei aplicații este prezentată în figura 22.1.

```
$ontext
Stationary flow of an incompressible fluid in a rectangular
area.
$offtext

set X /u1*u15/;
set Y /u1*u20/;
set Yout(Y) /u5*u16/;

Set Inside(X,Y);
```

```
    Inside(X,Y)                   = yes;
    Inside(X,Y)$(ord(X)=1)        = no;
    Inside(X,Y)$(ord(X)=card(X)) = no;
    Inside(X,Y)$(ord(Y)=1)        = no;
    Inside(X,Y)$(ord(Y)=card(Y)) = no;

scalar dx     /1/;
scalar dy     /1/;
scalar r      /1000/;
parameter m(X,Y);
          m(X,Y) := 0.5;

variables
        obj        objective
        D(X,Y)     error
        P(X,Y)     pressure
        Vx(X,Y)    x-direction velocity
        Vy(X,Y)    y-direction velocity
        Vdx(Y)     delta Vx at the outlet;

* Variable bounds
  Vx.up(X,Y)    =  1.5;
  Vx.lo(X,Y)    = -1.5;
  Vx.l(X,Y)     =  0.5;

  Vy.up(X,Y)    =  1.0;
  Vy.lo(X,Y)    = -1.0;
  Vy.l(X,Y)     =  0.0;

  D.lo(X,Y)     =  0.0;
  D.up(X,Y)     =  0.0000005;

  Vy.l(X,Y)     =  0.0;
  Vy.l(X,Y)$(inside(X,Y)) = 0.0000001;

  P.up(X,Y)     =  2000;
  P.lo(X,Y)     = -2000;
  P.l(X,Y)      =  0.000001;

*Boundary conditions
         Vx.lo('u1',Y)   =0.5;
         Vx.fx('u15',Y)  =0.5;
         Vx.fx(X,'u1')   = 0;
         Vx.fx(X,'u20')  = 0;
         Vx.fx('u1','u2') = 0;
         Vx.fx('u1','u3') = 0;
         Vx.fx('u1','u4') = 0;
         Vx.fx('u1','u17') = 0;
         Vx.fx('u1','u18') = 0;
         Vx.fx('u1','u19') = 0;
         Vx.fx('u15','u2')=0;
         Vx.fx('u15','u3')=0;
         Vx.fx('u15','u4')=0;
         Vx.fx('u15','u17')=0;
         Vx.fx('u15','u18')=0;
         Vx.fx('u15','u19')=0;
```

```
          Vy.fx('u1',Y) = 0;
          Vy.fx(X,'u1') = 0;
          Vy.fx(X,'u20') =0;
          Vy.fx('u15',Y)=0;

Equations
          For_Vx(X,Y)
          For_Vy(X,Y)
          Div_Vxy(X,Y)
          Vx_Vx(Y)
          Ben ;

For_Vx(X,Y)$(Inside(X,Y))..
   (P(X+1,Y)-P(X,Y))/(r*dx) =e=
     m(x,Y)*((Vx(X+1,Y)-2*Vx(X,Y)+Vx(x-1,Y))/(dx*dy)
    +         (Vx(X,Y+1)-2*Vx(X,Y)+Vx(X,Y-1))/(dx*dy));

For_Vy(X,Y)$(Inside(X,Y))..
   (P(X,Y+1)-P(X,Y))/(r*dy) =e=
     m(X,Y)*((Vy(X+1,Y)-2*Vy(X,Y)+Vy(X-1,Y))/(dx*dy)
    +         (Vy(X,Y+1)-2*Vy(X,Y)+Vy(X,Y-1))/(dy*dx));

Div_Vxy(X,Y)$((ord(X) > 1) $ (ord(Y) > 1))..
    (Vx(X-1,Y)-Vx(X,Y))/dx + (Vy(X,Y-1)-Vy(X,Y))/dy =e=
          D(X,Y);

Vx_Vx(Y).. Vdx(Y) =e= Vx('u15',Y)-Vx('u14',Y);

Ben.. obj =e= SUM(Y$Yout(Y), Vdx(Y)*Vdx(Y))
              +SUM((X,Y),(D(X,Y)*D(X,Y)));

Model flow /All/;
$onecho >bench.opt
  solvers conopt knitro minos
$offecho
flow.optfile=1;
option nlp=bench;

Solve flow using nlp minimizing obj;

* Put the solution
  file res1 /pressure.dat/
  put res1;
  loop(X, loop(Y, put P.l(x,y):9:1; ); put /;); put /;
  file res2 / vx.dat/
  put res2;
  loop(X, loop(Y, put Vx.l(x,y):9:1; ); put /;); put /;
  file res3 / vy.dat/
  put res3;
  loop(X, loop(Y, put Vy.l(x,y):5:1; ); put /;); put /;
  file res4 /err.dat/
  put res4
  loop(X, loop(Y, put D.l(x,y):5:1; ); put /;); put /;
* End Flow
```

Fig. 22.1. Expresia GAMS a aplicației G22 (Flow).

În tabelul 22.1 se prezintă performanțele algoritmilor CONOPT, KNITRO şi MINOS.

Tabelul 22.1

$n = 1183$ (variabile), $m = 755$ (restricții)

	#iter	**time**	**vfo**
CONOPT	5	0.098	0
KNITRO	9	0.430	0
MINOS	997	0.602	0

Dacă vizualizăm eroarea de aproximare prin diferențe finite, vom vedea că discretizarea modelului de mai sus nu introduce nici o eroare. De fapt şi valoarea optimă a funcției obiectiv furnizată de cei trei algoritmi de optimizare este nulă. Figura 22.2 prezintă evoluția vitezei în direcția x.

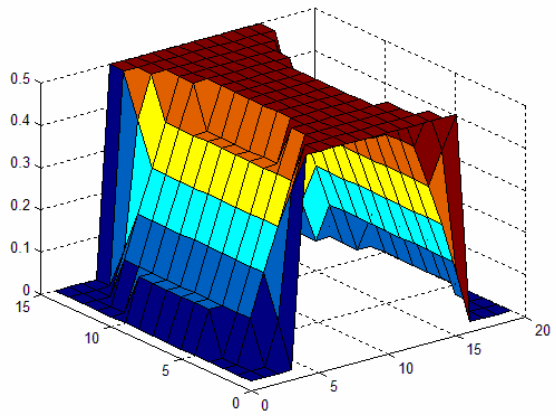

Fig. 22.2. Viteza în direcția x.

Figura 22.3 prezintă evoluția presiunii.

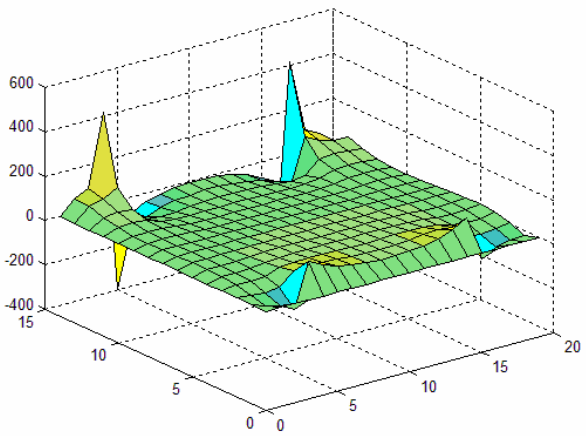

Fig. 22.3. Presiunea.

Aplicaţia G23 (*Robust*)
Stabilitatea robustă a unui sistem liniar. Determinarea în spaţiul parametrilor a celei mai mari regiuni posibile pentru care sistemul rămâne stabil la variaţia parametrilor.
Analiza stabilităţii robuste a sistemelor liniare constă în identificarea în spaţiul parametrilor a celei mai mari regiuni posibile pentru care sistemul rămâne stabil la variaţia parametrilor. Aceasta este determinată de rădăcinile polinomului caracteristic a sistemului în buclă închisă. Într-adevăr, pentru sistemul liniar dinamic din figura 23.1, în care $H(s,q)$ şi $C(s,q)$ sunt funcţiile de transfer ale instalaţiei şi respectiv a regulatorului, polinomul caracteristic este

$$P(s,q) = \det(I + H(s,q)C(s,q)),$$

unde $q \in Q$ este vectorul parametrilor necunoscuţi, care în cazul cel mai general, intervin atât în instalaţie cât şi în regulator.

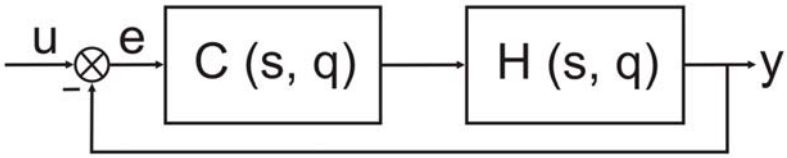

Fig. 23.1. Funcţiile de transfer ale instalaţiei şi regulatorului.

După dezvoltarea determinantului, ecuaţia caracteristică apare sub forma:

$$P(s,q) = a_n(q)s^n + a_{n-1}(q)s^{n-1} + \cdots + a_1(q)s + a_0(q) = 0,$$

unde $a_i(q)$, $i = 0,\ldots,n$, sunt funcţii polinomiale de q. Cu acestea, marginea de stabilitate k_m se defineşte ca:

$$k_m(j\omega) = \inf\{k : P(j\omega, q(k) = 0, \forall q \in Q\}.$$

Stabilitatea robustă a sistemului este garantată dacă şi numai dacă $k_m \geq 1$. Observăm că în situaţiile reale incertitudinea în parametrii q cel mai simplu se descrie ca deviaţii pozitive şi negative faţă de o anumită valoare nominală a parametrilor. În acest caz, testarea stabilităţii robuste a unui sistem cu polinomul caracteristic $P(j\omega,q)$ implică rezolvarea următoarei probleme de optimizare neconvexă [Floudas *et al.*, 1999]:

$$\min_{q_i,k,\omega\geq 0} k \tag{23.1}$$

referitor la:

$$\mathrm{Re}\left[P(j\omega,q)\right] = 0,$$
$$\mathrm{Im}\left[P(j\omega,q)\right] = 0,$$
$$q_i^N - \Delta q_i^- k \leq q_i \leq q_i^N + \Delta q_i^+ k, \quad i = 1,\ldots,n_q,$$

unde $q^N = [q_1^N \cdots q_{n_q}^N]$ este un punct din spațiul parametrilor pentru care sistemul în buclă închisă este stabil, iar $\Delta q^- = [\Delta q_1^- \cdots \Delta q_{n_q}^-]$ şi $\Delta q^+ = [\Delta q_1^+ \cdots \Delta q_{n_q}^+]$ sunt vectorii estimațiilor marginilor între care se presupune că parametrii q_i, $i = 1, \ldots, n_q$, aparțin.

Exemplul care urmează a fost propus de Gaston şi Safonov [1988] şi analizat de Psarris şi Floudas [1995]. În figura 23.2. se arată funcțiile de transfer ale sistemului în buclă închisă.

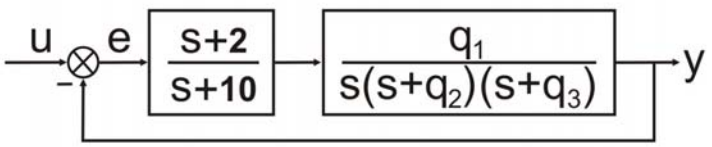

Fig. 23.2. Sistemul liniar în buclă închisă.

Pentru acest sistem

$$y = \frac{q_1(s+2)}{s(s+q_2)(s+q_3)(s+10) + q_1(s+2)} u.$$

Polinomul caracteristic este

$$P(s,q) = s^4 + (10 + q_2 + q_3)s^3 + (10q_2 + 10q_3 + q_2q_3)s^2 + (10q_2q_3 + q_1)s + 2q_1.$$

Cu acestea problema (23.1) care determină valoarea marginii de stabilitate este:

$$\min k$$

referitor la:

$$\begin{aligned}
&\omega^4 - (10q_2 + 10q_3 + q_2q_3)\omega^2 + 2q_1 = 0,\\
&-(10 + q_2 + q_3)\omega^3 + (10q_2q_3 + q_1)\omega = 0,\\
&q_1^N - \Delta q_1^- k \le q_1 \le q_1^N + \Delta q_1^+ k,\\
&q_2^N - \Delta q_2^- k \le q_2 \le q_2^N + \Delta q_2^+ k,\\
&q_3^N - \Delta q_3^- k \le q_3 \le q_3^N + \Delta q_3^+ k,\\
&\omega \ge 0,
\end{aligned}$$

unde $q^N = [800 \;\; 4 \;\; 6]$ şi $\Delta q^- = \Delta q^+ = [800 \;\; 2 \;\; 3]$.

Expresia GAMS a problemei de stabilitate de mai sus este arătată în figura 23.3.

```
$ONTEXT
 The robust stability of a linear system.
$OFFTEXT

Scalar q1N       /800/
       q2N       /4/
       q3N       /6/
       d1        /800/
```

```
        d2        /2/
        d3        /3/ ;

VARIABLES
        q1
        q2
        q3
        k
        z
        omega;

POSITIVE VARIABLES q1, q2, q3, k, omega;

EQUATIONS
          fobj                        Objective function
          g1, g2,
          b1l, b1u,  b2l,b2u, b3l,b3u;


fobj..  z =E= k;

g1  .. POWER(omega,4) - (Power(omega,2))*(10*q3+10*q2+q2*q3) +
                   2*q1 =e= 0;

g2  .. -(POWER(omega,3))*(10+q2+q3) + omega*(10*q2*q3+q1)
                         =e= 0;

b1l  .. q1N-d1*k =l= q1;
b1u  .. q1 =l= q1N+d1*k;

b2l  .. q2N-d2*k =l= q2;
b2u  .. q2 =l= q2N+d2*k;

b3l  .. q3N-d3*k =l= q3;
b3u  .. q3 =l= q3N+d3*k;

*Initial point
q1.l= 100;
q2.l= 4;
q3.l= 8;
k.l = 6;
omega.l = 10;

MODEL robust /ALL/;
SOLVE robust USING NLP MINIMIZING z;
* End Robust
```

Fig. 23.3. Expresia GAMS a aplicației G23 (Robust).

Soluția acestei probleme este:

$$q_1 = 1073.392 \qquad k = 0.342$$
$$q_2 = 3.317 \qquad \omega = 8.228.$$
$$q_3 = 4.975$$

Deoarece $k < 1$ rezultă că incertitudinile descrise de marginile de mai sus conțin puncte instabile. Cu alte cuvinte, soluția problemei ne furnizează marginea de stabilitate k, frecvența ω şi cea mai defavorabilă combinație de parametri pentru care apare instabilitatea. Performanțele algoritmilor CONOPT, KNITRO şi MINOS sunt ilustrate în tabelul 23.1.

Tabelul 23.1

$n = 6$ (variabile), $m = 9$ (restricții)

	#iter	**time**	**vfo**
CONOPT	38	0.063	0.341739553
KNITRO	10	0.100	0.341739613
MINOS	41	0.367	0.341739611

Dacă considerăm acum $\Delta q^- = \Delta q^+ = [8\ 1\ 1]$, atunci obținem marginea de stabilitate $k = 1.395$ şi frecvența $\omega = 7.366$, care arată că pentru acest domeniu de valori ale parametrilor sistemul este robust stabil.

Aplicația G24 (*Benz*)

Analiza stabilității autobuzului Daimler-Benz 0305. Determinarea în spațiul parametrilor a celei mai mari regiuni posibile pentru care autobuzul rămâne stabil la variația parametrilor.

Autobuzul Daimler-Benz 0305 ghidat de la distanță, cu intrarea δ ca unghiul de atac furnizată de regulator şi ieşirea y ca poziția antenei, are următoarea funcție de transfer [Ackermann, *et al.* 1991]:

$$H(s, q_1, q_2) = \frac{609.8 q_1^2 q_2 s^2 + 388600 q_1 s + 48280 q_1^2}{s^3 (q_1^2 q_2^2 s^2 + 1077 q_1 q_2 s + 16.8 q_1 q_2 + 270000)},$$

unde parametrii sistemului sunt: q_1 este viteza autobuzului şi $q_2 = m/f$ este raportul dintre masa autobuzului m şi coeficientul de frecare f. Regulatorul are funcția de transfer:

$$C(s) = \frac{2344 s^2 + 10938 s + 9375}{s^3 + 50 s^2 + 1250 s + 15625}.$$

Considerăm că $q^N = [17.5\ \ 20]$ este un punct din spațiul parametrilor pentru care sistemul în buclă închisă este stabil, iar $\Delta q^- = \Delta q^+ = [14.5\ \ 15]$ sunt vectorii estimațiilor marginilor între care se presupune că parametrii q_1 şi q_2 aparțin. Dorim să calculăm marginea de stabilitate pentru acest interval de variație a parametrilor. Calculând polinomul caracteristic în buclă închisă pentru regulatorul considerat, atunci problema (23.1) care determină valoarea marginii de stabilitate este următoarea:

$$\min k$$

referitor la:

$$a_8(q)\omega^8 - a_6(q)\omega^6 + a_4(q)\omega^4 - a_2(q)\omega^2 + a_0(q) = 0,$$
$$a_7(q)\omega^6 - a_5(q)\omega^4 + a_3(q)\omega^2 - a_1(q) = 0,$$
$$17.5 - 14.5k \le q_1 \le 17.5 + 14.5k,$$
$$20 - 15k \le q_2 \le 20 + 15k,$$

unde:

$$a_0(q) = 453\,10^6 q_1^2,$$
$$a_1(q) = 528\,10^6 q_1^2 + 3640\,10^6 q_1,$$
$$a_2(q) = 5.27\,10^6 q_1^2 q_2 + 113\,10^6 q_1^2 + 4250\,10^6 q_1,$$
$$a_3(q) = 6.93\,10^6 q_1^2 q_2 + 911\,10^6 q_1 + 4220\,10^6,$$
$$a_4(q) = 1.45\,10^6 q_1^2 q_2 + 16.8\,10^6 q_1 q_2 + 338\,10^6,$$
$$a_5(q) = 15.6\,10^3 q_1^2 q_2^2 + 840 q_1^2 q_2 + 1.35\,10^6 q_1 q_2 + 13.5\,10^6,$$
$$a_6(q) = 1.25\,10^3 q_1^2 q_2^2 + 16.8 q_1^2 q_2 + 53.9\,10^3 q_1 q_2 + 270\,10^3,$$
$$a_7(q) = 50 q_1^2 q_2^2 + 1080 q_1 q_2,$$
$$a_8(q) = q_1^2 q_2^2.$$

Expresia GAMS a acestei aplicații este prezentată în figura 24.1.

```
$ONTEXT
Stability of a wire guided Daimler-Benz 0305 bus.
$OFFTEXT

VARIABLES
         q1
         q2
         w   frequency
         k   stability margin
    objval   objective function variable;

FREE VARIABLES    objval;

EQUATIONS
         f Objective function
         g1
         g2
         b1l, b1u, b2l, b2u;

f  .. objval =e=k;

g1  .. (POWER(q1,2)*POWER(q2,2))*POWER(w,8) -
       (1.25*1000*POWER(q1,2)*POWER(q2,2) + 16.8*POWER(q1,2)*q2 +
        53.9*1000*q1*q2 + 270*1000)*POWER(w,6) +
       (1.45*POWER(10,6)*POWER(q1,2)*q2 + 16.8*POWER(10,6)*q1*q2 +
        POWER(10,6)*338)*POWER(w,4) -
       (5.72*POWER(10,6)*POWER(q1,2)*q2 + 113*POWER(10,6)*POWER(q1,2) +
        4250*POWER(10,6)*q1)*POWER(w,2) +
       (453*POWER(10,6)*POWER(q1,2)) =e= 0;

g2  .. (50*POWER(q1,2)*POWER(q2,2) + 1080*q1*q2)*POWER(w,6) -
```

```
        (15.6*1000*POWER(q1,2)*POWER(q2,2) + 840*POWER(q1,2)*q2 +
         1.35*POWER(10,6)*q1*q2 + POWER(10,6)*13.5)*POWER(w,4) +
        (6.93*POWER(10,6)*POWER(q1,2)*q2 + 911*POWER(10,6)*q1 +
         POWER(10,6)*4220)*POWER(w,2) -
        (528*POWER(10,6)*POWER(q1,2) + 3640*POWER(10,6)*q1) =e= 0;

b1l  .. 17.5 - 14.5*k =l= q1;
b1u  .. q1 =l= 17.5 + 14.5*k;
b2l  .. 20.0 - 15.0*k =l= q2;
b2u  .. q2 =l= 20.0 + 15.0*k;

* Bounds
q1.LO=0;
q1.UP=2;
q2.LO=0;
q2.UP=2;
w.LO =0;
w.UP =2;
k.LO =0;
k.UP =2;

* Initial values;
q1.l = 0.1;
q2.l = 0.1;
w.l  = 0.1;
k.l  = 0.1;

MODEL benz /ALL/;
benz.optfile=1;
benz.workspace=120;
$onecho >bench.opt
  solvers conopt knitro minos
$offecho
option nlp=bench;

SOLVE benz USING NLP MINIMIZING objval;
* End Benz
```

Fig. 24.1. Expresia GAMS a aplicației G24 (Benz).

Tabelul 24.1 conține performanțele algoritmilor utilizați în acest studiu.

Tabelul 24.1

$n = 5$ (variabile), $m = 7$ (restricții)

	#iter	time	vfo
CONOPT	12	0.090	1.206896
KNITRO	233	0.340	1.206896
MINOS	34	0.160	1.206897

Soluția acestei aplicații este următoarea:

$$q_1 = 1.6653e-6, \quad q_2 = 1.897, \quad \omega = 0.001, \quad k = 1.207.$$

Deoarece $k > 1$ rezultă că pentru acest domeniu de valori ale parametrilor şi frecvența $\omega = 0.001$ sistemul este robust stabil. Dacă mărim domeniul estimațiilor marginilor între care se presupune că parametrii q_1 şi q_2 evaluează la valorile $\Delta q^- = \Delta q^+ = [18.5\ 17]$, atunci marginea de stabilitate se reduce la $k = 1.059$ la aceeaşi frecvență.

Aplicația G25 (*Fiat*)
În această aplicație considerăm *analiza de stabilitate a instalației de aprindere prin scânteie (bujie) a automobilului Fiat Dedra.*
Considerând un punct din spațiul parametrilor pentru care sistemul în buclă închisă este stabil, şi anumite estimații ale marginilor între care se presupune că parametrii $q_i,\ i=1,\ldots,7,$ aparțin, atunci determinarea marginii de stabilitate pentru intervalul de variație a parametrilor dat se face prin rezolvarea următoarei probleme [Abate *et al*., 1994], [Barmish, 1994].

$$\min k$$

referitor la:

$$\begin{aligned}
&-a_6(q)\omega^6 + a_4(q)\omega^4 - a_2(q)\omega^2 + a_0(q) = 0,\\
&a_7(q)\omega^6 - a_5(q)\omega^4 + a_3(q)\omega^2 - a_1(q) = 0,\\
&3.4329 - 1.02721k \le q_1 \le 3.4329 + 1.02721k,\\
&0.1627 - 0.06k \le q_2 \le 0.1627 + 0.06k,\\
&0.1139 - 0.0782k \le q_3 \le 0.1139 + 0.0782k,\\
&1.2539 - 0.3068k \le q_4 \le 1.2539 + 0.3068k,\\
&0.0208 - 0.0108k \le q_5 \le 0.0208 + 0.0108k,\\
&5.0247 - 2.4715k \le q_6 \le 5.0247 + 2.4715k,\\
&1 - 2k \le q_7 \le 1 + 2k,
\end{aligned}$$

unde:

$$\begin{aligned}
a_0(q) &= 6.82079\,10^{-5} q_1 q_3 q_4^2 + 6.82079\,10^{-5} q_1 q_2 q_4 q_5,\\
a_1(q) &= 7.6176\,10^{-4} q_2^2 q_5^2 + 7.6176\,10^{-4} q_3^2 q_4^2 + 4.02141\,10^{-4} q_1 q_2 q_5^2 +\\
&\quad 0.00336706 q_1 q_3 q_4^2 + 6.82079\,10^{-5} q_1 q_4 q_5 + 5.16120\,10^{-4} q_2^2 q_5 q_6 +\\
&\quad 0.00336706 q_1 q_2 q_4 q_5 + 6.82079\,10^{-5} q_1 q_2 q_4 q_7 + 6.28987\,10^{-5} q_1 q_2 q_5 q_6 +\\
&\quad 4.02141\,10^{-4} q_1 q_3 q_4 q_5 + 6.28987\,10^{-5} q_1 q_3 q_4 q_6 + 0.00152352 q_2 q_3 q_4 q_5 +\\
&\quad 5.1612\,10^{-4} q_2 q_3 q_4 q_6,\\
a_2(q) &= 4.02141\,10^{-4} q_1 q_5^2 + 0.00152352 q_2 q_5^2 + 0.0552 q_2^2 q_5^2 + 0.0552 q_3^2 q_4^2 +\\
&\quad 0.0189477 q_1 q_2 q_5^2 + 0.034862 q_1 q_3 q_4^2 + 0.00336706 q_1 q_4 q_5 +\\
&\quad 6.82079\,10^{-5} q_1 q_4 q_7 + 6.28987\,10^{-5} q_1 q_5 q_6 + 0.00152352 q_3 q_4 q_5 +\\
&\quad 5.1612\,10^{-4} q_3 q_4 q_6 - 0.00234048 q_3^2 q_4 q_6 + 0.034862 q_1 q_2 q_4 q_5 +\\
&\quad 0.0237398 q_2^2 q_5 q_6 + 0.00152352 q_2^2 q_5 q_7 + 5.1612\,10^{-4} q_2^2 q_6 q_7 +\\
&\quad 0.00336706 q_1 q_2 q_4 q_7 + 0.00287416 q_1 q_2 q_5 q_6 + 8.04282\,10^{-4} q_1 q_2 q_5 q_7 +\\
&\quad 6.28987\,10^{-5} q_1 q_2 q_6 q_7 + 0.0189477 q_1 q_3 q_4 q_5 + 0.00287416 q_1 q_3 q_4 q_6 +\\
&\quad 4.02141\,10^{-4} q_1 q_3 q_4 q_7 + 0.1104 q_2 q_3 q_4 q_5 + 0.0237398 q_2 q_3 q_4 q_6 +\\
&\quad 0.00152352 q_2 q_3 q_4 q_7 - 0.00234048 q_2 q_3 q_5 q_6 + 0.00103224 q_2 q_5 q_6,
\end{aligned}$$

$$
\begin{aligned}
a_3(q) = {} & 0.0189477 q_1 q_5^2 + 0.1104 q_2 q_5^2 + 5.1612\,10^{-4} q_5 q_6 + q_2^2 q_5^2 + \\
& 7.6176\,10^{-4} q_2^2 q_7^2 + q_3^2 q_4^2 + 0.1586 q_1 q_2 q_5^2 + 4.0214\,110^{-4} q_1 q_2 q_7^2 + \\
& 0.0872 q_1 q_3 q_4^2 + 0.034862 q_1 q_4 q_5 + 0.00336706 q_1 q_4 q_7 + \\
& 0.00287416 q_1 q_5 q_6 + 6.28987\,10^{-5} q_1 q_6 q_7 + 0.00103224 q_2 q_6 q_7 + \\
& 0.1104 q_3 q_4 q_5 + 0.0237398 q_3 q_4 q_6 + 0.00152352 q_3 q_4 q_7 - \\
& 0.00234048 q_3 q_5 q_6 + 0.1826 q_2^2 q_5 q_6 + 0.1104 q_2^2 q_5 q_7 + \\
& 0.0237398 q_2^2 q_6 q_7 - 0.0848 q_3^2 q_4 q_6 + 0.0872 q_1 q_2 q_4 q_5 + \\
& 0.034862 q_1 q_2 q_4 q_7 + 0.0215658 q_1 q_2 q_5 q_6 + 0.0378954 q_1 q_2 q_5 q_7 + \\
& 0.00287416 q_1 q_2 q_6 q_7 + 0.1586 q_1 q_3 q_4 q_5 + 0.0215658 q_1 q_3 q_4 q_6 + \\
& 0.0189477 q_1 q_3 q_4 q_7 + 2 q_2 q_3 q_4 q_5 + 0.1826 q_2 q_3 q_4 q_6 + 0.1104 q_2 q_3 q_4 q_7 - \\
& 0.0848 q_2 q_3 q_5 q_6 - 0.00234048 q_2 q_3 q_6 q_7 + 7.6176\,10^{-4} q_5^2 + \\
& 0.0474795 q_2 q_5 q_6 + 8.04282\,10^{-4} q_1 q_5 q_7 + 0.00304704 q_2 q_5 q_7, \\
a_4(q) = {} & 0.1586 q_1 q_5^2 + 4.0214\,110^{-4} q_1 q_7^2 + 2 q_2 q_5^2 + 0.00152352 q_2 q_7^2 + \\
& 0.0237398 q_5 q_6 + 0.00152352 q_5 q_7 + 5.1612\,10^{-4} q_6 q_7 + \\
& 0.0552 q_2^2 q_7^2 + 0.01898477 q_1 q_2 q_7^2 + 0.0872 q_1 q_4 q_5 + \\
& 0.034862 q_1 q_4 q_7 + 0.0215658 q_1 q_5 q_6 + 0.00287416 q_1 q_6 q_7 + \\
& 0.0474795 q_2 q_6 q_7 + 2 q_3 q_4 q_5 + 0.1826 q_3 q_4 q_6 + 0.1104 q_3 q_4 q_7 - \\
& 0.0848 q_3 q_5 q_6 - 0.00234048 q_3 q_6 q_7 + 2 q_2^2 q_5 q_7 + 0.1826 q_2^2 q_6 q_7 + \\
& 0.0872 q_1 q_2 q_4 q_7 + 0.3172 q_1 q_2 q_5 q_7 + 0.0215658 q_1 q_2 q_6 q_7 + \\
& 0.1586 q_1 q_3 q_4 q_7 + 2 q_2 q_3 q_4 q_7 - 0.0848 q_2 q_3 q_6 q_7 + 0.0552 q_5^2 + \\
& 0.3652 q_2 q_5 q_6 + 0.0378954 q_1 q_5 q_7 + 0.2208 q_2 q_5 q_7, \\
a_5(q) = {} & 0.0189477 q_1 q_7^2 + 0.11104 q_2 q_7^2 + 0.1826 q_5 q_6 + 0.1104 q_5 q_7 + \\
& 0.0237398 q_6 q_7 + q_2^2 q_7^2 + 0.1586 q_1 q_2 q_7^2 + 0.0872 q_1 q_4 q_7 + \\
& 0.0215658 q_1 q_6 q_7 + 0.3652 q_2 q_6 q_7 + 2 q_3 q_4 q_7 - 0.0848 q_3 q_6 q_7 + \\
& q_5^2 + 7.6176\,10^{-4} q_7^2 + 0.3172 q_1 q_5 q_7 + 4 q_2 q_5 q_7, \\
a_6(q) = {} & 0.1586 q_1 q_7^2 + 2 q_2 q_7^2 + 2 q_5 q_7 + 0.1826 q_6 q_7 + 0.0552 q_7^2, \\
a_7(q) = {} & q_7^2.
\end{aligned}
$$

Expresia GAMS a acestei aplicații este prezentată în figura 25.1.

```
$ONTEXT
Analysis of the stability margin of the spark ignition engine
Fiat Dedra.
$OFFTEXT

VARIABLES
         q1, q2, q3, q4, q5, q6, q7
         w   frequency
```

```
         k   stability margin
         a0, a1, a2, a3, a4, a5, a6, a7
     objval    objective function variable;

FREE VARIABLES    objval;

EQUATIONS
         f Objective function
         g1
         g2
         b1l, b1u
         b2l, b2u
         b3l, b3u
         b4l, b4u
         b5l, b5u
         b6l, b6u
         b7l, b7u
         ga0, ga1, ga2, ga3, ga4, ga5, ga6, ga7  ;

f  .. objval =e=k;

g1  .. -a6*POWER(w,6) + a4*POWER(w,4) - a2*POWER(w,2) + a0 =e= 0;
g2  .. a7*POWER(w,6) - a5*POWER(w,4) + a3*POWER(w,2) - a1 =e= 0;

b1l  .. 3.4329-1.02721*k =l= q1;
b1u  .. q1 =l= 3.4320+1.02721*k;
b2l  .. 0.1627-0.06*k =l= q2;
b2u  .. q2 =l= 0.1627+0.06*k;
b3l  .. 0.1139-0.0782*k =l= q3;
b3u  .. q3 =l= 0.1139+0.0782*k;
b4l  .. 1.2539-0.3068*k =l= q4;
b4u  .. q4 =l= 1.2539+0.3068*k;
b5l  .. 0.0208-0.0108*k =l= q5;
b5u  .. q5 =l= 0.0208+0.08*k;
b6l  .. 5.0247-2.4715*k =l= q6;
b6u  .. q6 =l= 5.0247+2.4715*k;
b7l  .. 1.0-2*k =l= q7;
b7u  .. q7 =l= 1.0+2*k;

ga0  .. a0 =e= 6.82079e-05*q1*q3*POWER(q4,2) + 6.82079e-05*q1*q2*q4*q5;

ga1  .. a1 =e= 0.00076176*POWER(q2,2)*POWER(q5,2) +
               0.00076176*POWER(q3,2)*POWER(q4,2) +
               0.000402141*q1*q2*POWER(q5,2) +
               0.00337606*q1*q3*POWER(q4,2) +
               6.82079e-05*q1*q4*q5 + 0.00051612*POWER(q2,2)*q5*q6 +
               0.00337606*q1*q2*q4*q5 + 6.82079e-05*q1*q2*q4*q7 +
               6.28987e-05*q1*q2*q5*q6 + 0.000402141*q1*q3*q4*q5 +
               6.28987e-05*q1*q3*q4*q6 + 0.00152352*q2*q3*q4*q5 +
               0.00051612*q2*q3*q4*q6;

ga2  .. a2 =e= 0.000402141*q1*POWER(q5,2) + 0.00152352*q2*POWER(q5,2) +
               0.0552*POWER(q2,2)*POWER(q5,2) +
               0.0552*POWER(q3,2)*POWER(q4,2) +
               0.0189477*q1*q2*POWER(q5,2)+0.034862*q1*q3*POWER(q4,2) +
               0.00336706*q1*q4*q5 + 6.82079e-05*q1*q4*q7 +
               6.28987e-05*q1*q5*q6 + 0.00152352*q3*q4*q5 +
               0.00051612*q3*q4*q6 - 0.00234048*POWER(q3,2)*q4*q6 +
               0.034862*q1*q2*q4*q5 + 0.0237398*POWER(q2,2)*q5*q6 +
               0.00152352*POWER(q2,2)*q5*q7 +
               0.00051612*POWER(q2,2)*q6*q7 +
               0.00336706*q1*q2*q4*q7 + 0.00287416*q1*q2*q5*q6 +
               0.000804282*q1*q2*q5*q7 + 6.28987e-05*q1*q2*q6*q7 +
```

```
                0.0189477*q1*q3*q4*q5 + 0.00287416*q1*q3*q4*q6 +
                0.000402141*q1*q3*q4*q7 + 0.1104*q2*q3*q4*q5 +
                0.0237398*q2*q3*q4*q6 + 0.00152352*q2*q3*q4*q7 -
                0.00234048*q2*q3*q5*q6 + 0.00103224*q2*q5*q6;

ga3  .. a3 =e= 0.189477*q1*POWER(q5,2) + 0.1104*q2*POWER(q5,2) +
                0.00051612*q5*q6 + POWER(q2,2)*POWER(q5,2) +
                0.00076176*POWER(q2,2)*POWER(q7,2) +
                POWER(q3,2)*POWER(q4,2) +
                0.1586*q1*q2*POWER(q5,2)+0.000402141*q1*q2*POWER(q7,2) +
                0.0872*q1*q3*POWER(q4,2) + 0.034862*q1*q4*q5 +
                0.00336706*q1*q4*q7 + 0.00287416*q1*q5*q6 +
                6.28987e-05*q1*q6*q7 + 0.00103224*q2*q6*q7 +
                0.1104*q3*q4*q5 +
                0.0237398*q3*q4*q6 + 0.00152352*q3*q4*q7 -
                0.00234048*q3*q5*q6 +
                0.1826*POWER(q2,2)*q5*q6 + 0.1104*POWER(q2,2)*q5*q7 +
                0.0237398*POWER(q2,2)*q6*q7 - 0.0848*POWER(q3,2)*q4*q6 +
                0.0872*q1*q2*q4*q5 + 0.034862*q1*q2*q4*q7 +
                0.0215658*q1*q2*q5*q6 + 0.0378954*q1*q2*q5*q7 +
                0.00287416*q1*q2*q6*q7 + 0.1586*q1*q3*q4*q5 +
                0.0215658*q1*q3*q4*q6 + 0.0189477*q1*q3*q4*q7 +
                2*q2*q3*q4*q5 + 0.1826*q2*q3*q4*q6 +0.1104*q2*q3*q4*q7 -
                0.0848*q2*q3*q5*q6 - 0.00234048*q2*q3*q6*q7 +
                0.00076176*POWER(q5,2) + 0.0474795*q2*q5*q6 +
                0.000804282*q1*q5*q7 + 0.00304704*q2*q5*q7;

ga4  .. a4 =e= 0.1586*q1*POWER(q5,2) + 0.000402141*q1*POWER(q7,2) +
                2*q2*POWER(q5,2) + 0.00152352*q2*POWER(q7,2) +
                0.0237398*q5*q6 +
                0.00152352*q5*q7 + 0.00051612*q6*q7 +
                0.0552*POWER(q2,2)*POWER(q7,2) +
                0.0189477*q1*q2*POWER(q7,2) +
                0.0872*q1*q4*q5 + 0.034862*q1*q4*q7+0.0215658*q1*q5*q6 +
                0.00287416*q1*q6*q7 + 0.0474795*q2*q6*q7 + 2*q3*q4*q5 +
                0.1826*q3*q4*q6 + 0.1104*q3*q4*q7 - 0.0848*q3*q5*q6 -
                0.00234048*q3*q6*q7 + 2*POWER(q2,2)*q5*q7 +
                0.1826*POWER(q2,2)*q6*q7 + 0.0872*q1*q2*q4*q7 +
                0.3172*q1*q2*q5*q7 + 0.0215658*q1*q2*q6*q7 +
                0.1586*q1*q3*q4*q7 + 2*q2*q3*q4*q7-0.0848*q2*q3*q6*q7 +
                0.0552*POWER(q5,2) + 0.3652*q2*q5*q6 +
                0.0378954*q1*q5*q7 + 0.2208*q2*q5*q7;

ga5  .. a5 =e= 0.0189477*q1*POWER(q7,2) + 0.1104*q2*POWER(q7,2) +
                0.1826*q5*q6 + 0.1104*q5*q7 + 0.0237398*q6*q7 +
                POWER(q2,2)*POWER(q7,2) + 0.1586*q1*q2*POWER(q7,2) +
                0.0872*q1*q4*q7 + 0.0215658*q1*q6*q7 + 0.3652*q2*q6*q7 +
                2*q3*q4*q7 - 0.0848*q3*q6*q7 + POWER(q5,2) +
                0.00076176*POWER(q7,2) + 0.3172*q1*q5*q7 + 4*q2*q5*q7;

ga6  .. a6 =e= 0.1586*q1*POWER(q7,2) + 2*q2*POWER(q7,2) + 2*q5*q7 +
                0.1826*q6*q7 + 0.0552*POWER(q7,2);

ga7  .. a7 =e= POWER(q7,2);

* Bounds
*q1.UP = 3.4329;
*q2.UP = 0.1627;
*q3.UP = 0.1139;
*q4.LO = 0.2539;
*q5.UP = 0.0208;
*q6.LO = 2.0247;
*q7.LO = 1;
```

```
w.LO = 0;
w.UP = 10;
k.LO = 0;
k.UP = 10;

* Initial point
q1.l=0.2;
q2.l=0.02;
q3.l=0.1;
q4.l=0.3;
q5.l=0;
q6.l=2;
q7.l=4.5;
w.l=0;
k.l=2;

MODEL fiat /ALL/;

$onecho >bench.opt
  solvers conopt minos
$offecho

fiat.optfile=1;
option nlp=bench;

SOLVE fiat USING NLP MINIMIZING objval;
* End Fiat
```

Fig. 25.1. Expresia GAMS a aplicației G25 (Fiat).

Tabelul 25.1 prezintă performanțele algoritmilor CONOPT şi MINOS. KNITRO nu poate rezolva problema.

Tabelul 25.1

$n = 18$ (variabile), $m = 25$ (restricții)

	#iter	time	vfo
CONOPT	15	0.082	1.4593669
MINOS	59	0.414	3.341965

Din tabelul 25.1 se vede imediat că pentru această estimație a marginilor între care se presupune că parametrii q_i, $i = 1,\ldots,7$, aparțin, $k > 1$, ceea ce probează stabilitatea sistemului de aprindere prin scânteie a automobilului Fiat Dedra.

Referințe

Abate, M., Barmish, B., Murillo-Sanchez C., Tempo. R., (1994) *Application of some new tools to robust stability analysis of spark ignition engines: A case study.* IEEE Trans. Contr. Syst. Tech, 2, 1994, pp. 22-30

Ackermann, J., Käsbauer, D., Münch, R., (1991) *Robust gamma-stability analysis in a plant parameter space.* Automatica, 27, 1991, pp.75-85.

Andrei, N., (1996a) *Computational experience with a modified penalti-barrier method for large-scale nonlinear constrained optimization.* Working Paper No. AMOL-96-1, Research Institute for Informatics, ICI, Bucharest, February 6, 1996

Andrei, N., (1996b) *Computational experience with a modified penalty-barrier method for large-scale nonlinear, equality and inequality constrained optimization.* Technical

Paper No. AMOL-96-2, Research Institute for Informatics, ICI, Bucharest, February 12, 1996.

Andrei, N., (2001) *Numerical examples solved with SPENBAR – modified penalty barrier method for large-scale nonlinear programming problems.* ICI Technical Report No. 1/2001, February 2001. (Manuscris în Biblioteca Academiei Române.)

Andrei, N., (2003) *Modele, Probleme de Test şi Aplicații de Programare Matematică*. Editura Tehnică, Bucureşti, 2003.

Andrei, N., (2004a) *Calculul temperaturii staționare într-o placă dreptunghiulară.* Raport Tehnic ICI, March 18, 2004. În WORKS-2004, Bucharest, 2004. (Manuscris în Biblioteca Academiei Române.)

Andrei, N., (2004b) *Model de creştere economică Ramsey.* Raport Tehnic ICI, 2004. În WORKS-2004, Bucharest, 2004. (Manuscris în Biblioteca Academiei Române.)

Andrei, N., (2004c) *Optimizarea formei unui lanț suspendat ca capete.* Raport Tehnic ICI, 2004. În WORKS-2004, Bucharest, 2004. (Manuscris în Biblioteca Academiei Române.)

Andrei, N., (2004d) *Optimizarea funcționării a două rezervoare (Aplicație GAMS).* Raport Tehnic ICI, 2004. În WORKS-2004, Bucharest, 2004. (Manuscris în Biblioteca Academiei Române.)

Andrei, N., (2005) *Optimizarea funcționării unui sistem de lacuri.* Raport Tehnic ICI, 2005. În WORKS-2005, Bucharest, 2005. (Manuscris în Biblioteca Academiei Române.)

Andrei, N., (2007) *SCALCG – Scaled conjugate gradient algorithms for unconstrained optimization.* Technical Report No. 17/2007, Research Institute for Informatics, Bucharest, March 30, 2007. (Manuscris în Biblioteca Academiei Române.)

Andrei, N., (2009) *Critica Rațiunii Algoritmilor de Optimizare fără Restricții.* Editura Academiei Române, Bucureşti, 2009.

Anițescu, M., Şerban, R., (1998) *A sparse superlinearly convergent SQP with applications to two-dimensional shape optimization.* Preprint ANL/MCS-P706-0198, Argonne National Laboratory, Argonne, Illinois, 1998.

Averick, B.M., Carter, R.G., Moré, J.J., Xue, G.-L., (1994) *The MINPACK-2 test problem collection.* Preprint MCS-P153-0694, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, Illinois, 1994.

Avriel, M., Williams, A.C., (1971) *An extension of geometric programming with applications in engineering optimization.* Journal of Engineering Mathematics, 5, 1971, pp. 187-194.

Barmish, B.R., (1994) *New tools for robustness of linear systems.* MacMillan Publishing Company, New York, 1994.

Barro, R., (1998) *Notes on growth accounting*. Harward University Technical Report, December 17, 1998.

Barro, R., (1999) *Ramsey meets Laibson in the neoclassical growth model*. Harward University Technical Report, September, 1999.

Barro, R., Sala-i-Martin, X., (1998) *Economic growth*, MIT Press, 1998.

Ben-Tal, A., Eiger, G., Gershovitz, V., (1994) *Global minimization by reducting the duality gap.* Mathematical Programming 63, 1994, pp. 193-212.

Bondarenko, A.S., Bortz, D.M., Moré, J.J., (1999) *COPS: Large-scale nonlinearly constrained optimization problems.* Technical Report ANL/MCS-TM-237, Argonne National Laboratory, Argonne, Illinois, September 1998, October 1999 (revision).

Bryson, A.E., (1999) *Dynamic optimization.* Addison-Wesley, New-York, 1999.

Bryson, A.E., Ho, Y., (1975) *Applied optimal control: Optimization, Estimation and Control.* John Wiley, New York, 1975.

Brooke, A., Kendrick, D., Meeraus, A., Raman, R., Rosenthal, R.E., (1998) *GAMS A user's guide*. GAMS Development Corporation, December 1998.

Bulirsch, R., Nerz, E., Pesch, H.J., von Stryk, O., (1993) *Combining direct and indirect methods in nonlinear optimal control: Range maximization of a hang glider.* În Optimal Control, R. Bulirsch, A. Miele, J. Stoer and K.H. Well (Eds.) Birkhäuser Verlag, 1993, pp.273-288.

Cesari, L., (1983) *Optimization – theory and applications*. Springer Verlag, Bonn, 1983.

De Gaston, R.R.E., Safonov, M.G., (1988) Exact calculation of the multiloop stability margin. IEEE Trans. Autom. Control, Ac-33, pp.156-171, 1988.

Dembo, R.S., (1976) *A set of geometric programming test problems and their solution.* Mathematical Programming, 10, 1976, pp.192-213.

Dolan, E.D., Moré, J.J., Munson, T.S., (2004) *Benchmarking optimization software with COPS 3.0.* Preprint ANL/MCS-TM-273, Argonne National Laboratory, Argonne, Illinois, February 2004.

Floudas, C.A., Ciric, A.R., (1988) *Global optimum issues on heat exchanger networks synthesis.* În Proceedings of the Third International Symposium on Process Systems Engineering, Sydney, Australia, 1988, p.104.

Floudas, C.A., Pardalos, P.M., Adjiman, C.S., Esposito, W.R., Gümüs, Z.H., Harding, S.T., Klepeis, J.L., Meyer, C.A., Schweiger, C.A., (1999) *Handbook of test problems in Local and global optimization.* Kluwer Academic Publishers, Dordrecht, 1999.

Glowinski, R., (1984) *Numerical Methods for Nonlinear Variational Problems.* Springer-Verlag, Berlin, 1984.

Graham, RL., (1975) *The largest small hexagon.* J. Combin. Th., 18, 1975, pp.165-170.

Harrison, G.W., (2001) *MEEN, modeling the environment-economy nexus.* http://theweb.badm.sc.edu/glenn/meen_mod.htm, 2001.

Haverly, C., (1978) *Studies of the behavior of recursion for the pooling problem.*ACM-SIGMAP Bulletin 25, 1978, pp.19-28.

Kalvelagen, E., (2003) *An elementary Ramsey growth model.* GAMS Technical Report, March 12, 2003.

Ken-Ichi Inada, (1963) *On a two-sector model of economic growth: comments and a generalization.* Review of Economic Studies 30, pp.119-127, 1963.

Lasdon, L., Waren, A, Sarkar, S., Palacios, F., (1979) *Solving the pooling problem using generalized reduced gradient and successive linear programming algorithms.* ACM-SIGMAP Bulletin, 26, 1979, pp.9-15.

Leiby, P., şi **Rubin, J.,** (1997) *Technical documentation of the transitional alternative fuels and vehicles (TAFV) model.* Technical Report, Oak Ridge National Laboratory, July 1997.

Manne, A., şi **Wene, C.,** (1992) *MARKAL-MACRO: A linked model for energy-economy analysis.* Technical Report BNL-47161, Brookhaven National Laboratory, 1992.

McKinney, D., Savitsky, A., (2003) *Basic optimization models for water and energy management*, The University of Texas at Austin Thechnical Report, 2003.

Mössner-Beigel, M., (1995) *Optimale Steuerung für Industrieroboter unter Berücksichtigung der Getriebebedingten Elastizität.* Diplomarbeit, Universität Heidelberg, Fakultät für Mathematik, November 1995.

Nordhaus, W.D. (1992) *An optimal transition path for controlling greenhouse gasses.* Science 258, 1315-1319, 1992.

Pedersen, T.M., (1999) *The Ramsey model of optimal economic growth*. Technical Report, Institute of Economics, University of Copenhagen, December 1999.

Psarris, P., Floudas, C.A., (1995) Robust stabilty of systems with real parametric uncertainty: A global optimization approach. International Journal of Robust and Nonlinear Control, vol.5, pp.699-717, 1995.

Ramsey, F.P., (1928) *A mathematical theory of saving*. Economic Journal, vol.38, no.152, 1928, pp.543-559.

Visweswaran, V., Floudas, C.A., (1996) *Computational results for an efficient implementation of the GOP algorithm and its variants.* În Grassmann I.E. (Ed.) *Gloobal Optimization in Engineering Design,* Kluwer Book Series in Nonconvex Optimization and its Applications. Chapter 4, Dordrecht, 1996.

Von Stryk, O., (1999) *User's guide for DIRCOL (Version 2.1): A direct collocation method for the numerical solution of optimal control problems.* Technical Report, Technische Universität München, 1999.

31 Ianuarie 2011

♦