ECP 2ème année
Semestre 4, Electif 9, Année 2011
Mr DUMAS et Mr BOULMEZAOUD

**NUMERICAL OPTIMIZATION AND APPLICATIONS**

**2011, March 25th, 3 hours**

---

*All Scilab programs must be sent at the adress* `laurent.dumas@uvsq.fr` *at the end of the exam with a file name of the type* `ECP2011-firstname-name.sci` *and with the subject ECP2011.*

*The problem is in english but candidates can give a copy in french if they prefer.*

## A. Theoretical part

In the sequel, when there is no confusion, we denote by $\mathbb{R}^n$ the space of all the colomun vectors of the form $(x_1, ..., x_n)^T$, $x_i \in \mathbb{R}$ for $1 \leq i \leq n$.
Consider a quadratic function $f : \mathbb{R}^n \to \mathbb{R}$ of the form

$$f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^T A\boldsymbol{x} - \boldsymbol{b}^T\boldsymbol{x} + c, \quad \text{for } \boldsymbol{x} \in \mathbb{R}^n, \tag{1}$$

where $A$ is a $n \times n$ symmetric definite positive matrix, $\boldsymbol{b} \in \mathbb{R}^n$ is a given column vector and $c$ a constant. We denote by $0 < \lambda_1 \leq ... \leq \lambda_n$ the eigenvalues of $A$ and $(\boldsymbol{v}_i)_{1\leq i\leq n}$ a corresponding orthonormal basis of eigenvectors of $A$, that is

$$A\boldsymbol{v}_i = \lambda_i \boldsymbol{v}_i \text{ and } \|\boldsymbol{v}_i\| = 1 \text{ for } 1 \leq i \leq n.$$

Here $\|.\|$ denotes the usual euclidian norm ($\|\boldsymbol{x}\| = \sqrt{\boldsymbol{x}^T\boldsymbol{x}}$). We set

$$\eta = \frac{\lambda_n}{\lambda_1} \geq 1.$$

**I.** 1. Show that

$$(\boldsymbol{x}^T A\boldsymbol{x}).(\boldsymbol{x}^T A^{-1}\boldsymbol{x}) \geq \|\boldsymbol{x}\|^4 \text{ for all } \boldsymbol{x} \in \mathbb{R}^n.$$

Indication: you can decompose $\boldsymbol{x}$ on the orthonormal basis $(\boldsymbol{v}_i)_{1\leq i\leq n}$.

2. Let $P = (A - \lambda_1 I)(A - \lambda_n I)A^{-1}$. Show that

$$\boldsymbol{x}^T P\boldsymbol{x} \leq 0 \text{ for all } \boldsymbol{x} \in \mathbb{R}^n.$$

Deduce that

$$\boldsymbol{x}^T A\boldsymbol{x} + \lambda_1\lambda_n \boldsymbol{x}^T A^{-1}\boldsymbol{x} \leq (\lambda_1 + \lambda_n)\|\boldsymbol{x}\|^2 \quad \text{for all } \boldsymbol{x} \in \mathbb{R}^n.$$

3. Deduce the inequality

$$\|\boldsymbol{x}\|^4 \leq (\boldsymbol{x}^T A\boldsymbol{x}).(\boldsymbol{x}^T A^{-1}\boldsymbol{x}) \leq \frac{(\lambda_1+\lambda_n)^2}{4\lambda_1\lambda_n}\|\boldsymbol{x}\|^4 \text{ for all } \boldsymbol{x}\in\mathbb{R}^n.$$

Indication: you can use the inequality $\alpha^2+\beta^2 \geq 2\alpha\beta$.

**II.** Our purpose here is to minimize $f$. In the method of the steepest descent we start from an arbitrary point $\boldsymbol{x}_0$ and we slide down to the minimum of $f$ by defining the sequence $(\boldsymbol{x}_k)_{k\geq 0}$ as follows

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{r}_k \text{ where } \boldsymbol{r}_k = \boldsymbol{b} - A\boldsymbol{x}_k = -\nabla f(\boldsymbol{x}_k),$$

and $\alpha_k$ is chosen to minimize $f$ along the direction of $\boldsymbol{r}_k$, that is

$$\alpha_k \text{ minimizes } f(\boldsymbol{x}_k + \alpha \boldsymbol{r}_k),\ \alpha \geq 0.$$

We define the error vector as follows

$$\boldsymbol{e}_k = \boldsymbol{x}_k - \boldsymbol{x},$$

where $\boldsymbol{x}$ is the unique solution of the linear system

$$A\boldsymbol{x} = \boldsymbol{b}.$$

1. Show that $\boldsymbol{r}_k = -A\boldsymbol{e}_k$.
2. Show that

$$\alpha_k = \frac{\boldsymbol{r}_k^T \boldsymbol{r}_k}{\boldsymbol{r}_k^T A \boldsymbol{r}_k}.$$

$$\boldsymbol{r}_{k+1} = \boldsymbol{r}_k - \alpha_k A\boldsymbol{r}_k,\ \boldsymbol{e}_{k+1} = \boldsymbol{e}_k + \alpha_k \boldsymbol{r}_k.$$

3. Show that

$$\boldsymbol{e}_{k+1}^T A \boldsymbol{e}_{k+1} = \omega_k \boldsymbol{e}_k^T A \boldsymbol{e}_k,$$

where

$$\omega_k = 1 - \frac{\|\boldsymbol{r}_k\|^4}{(\boldsymbol{r}_k A \boldsymbol{r}_k)(\boldsymbol{r}_k A^{-1} \boldsymbol{r}_k)}.$$

4. Deduce that

$$\|\boldsymbol{e}_k\| \leq \sqrt{\eta}\left(\frac{\eta-1}{\eta+1}\right)^k \|\boldsymbol{e}_0\|.$$

### B. Computational part

Starting from an existing code, the objective is to implement a new selection principle, called stochastic ranking in order to handle constraints in a different way.

**I. The initial script** .

The initial script is given in annex and can be downloaded at the following adress: `http://www.math.uvsq.fr/~dumas/ECP2011exam.sci`. It is a real valued genetic algorithm, with no elitism, aimed to solve the 'can problem' with a fixed-penalty method (that is design a cylindrical can having a minimal surface with a volume greater than 300ml).

1. Explain the definition of the cost function in the script.
2. Describe and justify the mutation operator. In what sense can we say it is an adaptative mutation?
3. Describe the crossover operator. What variant of this crossover could you propose?
4. Comment the two figures that are plotted after the execution of this script.

**II. The stochastic ranking** .

Stochastic ranking is a method that is described below to rank $\lambda$ individuals in order to take into account the constraints in the ranking but with some randomness.

```
1   I_j = j ∀ j ∈ {1,...,λ}
2   for i = 1 to λ do
3       for j = 1 to λ − 1 do
4           sample u ∈ U(0,1) (uniform random number generator)
5           if (φ(x_{I_j}) = φ(x_{I_{j+1}}) = 0) or (u < P_f) then
6             if f(x_{I_j}) > f(x_{I_{j+1}}) then
7               swap(I_j, I_{j+1})
8             fi
9           else
10            if φ(x_{I_j}) > φ(x_{I_{j+1}}) then
11              swap(I_j, I_{j+1})
12            fi
13          fi
14      od
15      if no swap done break fi
    od
```

Fig. 2. Stochastic ranking procedure, $P_f = 0.45$.

In this algorithm, $f$ is the cost function that needs to be minimized (for instance in the can problem, the surface) and $\Phi$ is the penalty term (for instance, $\min(V-300,0)^2$ for the can problem). In this algorithm, the fixed parameter $P_f$ is responsible for the randomness in the ranking.

1. Describe the stochastic ranking when $P_f = 0$, respectively 1.
2. Implement (with Scilab or Matlab/Octave) the stochastic ranking of $\lambda$ elements with a given function $f$ and a penalty function $\Phi$.

**III The can problem** .

1. Compute the exact solution of the can problem.
2. Apply the stochastic ranking method in order to solve numerically the can problem. In the original script, only the functions 'evaluation' and 'selection' need to be modified. Compare the results with those from the inital script.

```
function Xeval=evaluation(X);
 mu=1;
 [Npop,n2]=size(X);
 Xeval=X;
  for i=1:Npop;
   penal=min(%pi*X(i,1)^2*X(i,2)/4-300,0)^2;
   Xeval(i,$)=%pi*X(i,1)^2/4+%pi*X(i,1)*X(i,2)+mu*penal;
  end
endfunction
////////////////////////////////
function bestX=best(X);
[Npop,n2]=size(X);
[y,k]=gsort(-X(:,$));
bestX=X(k(1),:);
endfunction
////////////////////////////////
function Xsel=selection(X)
[Npop,n2]=size(X);
[y,k]=gsort(X(:,$))
X=X(k,:);
p=1:Npop;
roulette=cumsum(p)/sum(p);
Xsel=[];
for i=1:Npop
  u=rand();isel=1;
  while (u>roulette(isel))
     isel=isel+1;
  end
  Xsel=[Xsel;X(isel,:)];
end
endfunction
////////////////////////////////////
function xcrois=croisement(x)
xcrois=x;
u=rand();
xcrois(1,:)=u*x(1,:)+(1-u)*x(2,:);
xcrois(2,:)=(1-u)*x(1,:)+u*x(2,:);
endfunction
////////////////////////////////////
function xmut=mutation(x,gen,Ngen,xmin,xmax)
bet=5;
if (rand()<1/2) then
   xmut=x+rand()*(xmax-x)*(1-(gen-1)/Ngen)^bet;
else
   xmut=x-rand()*(x-xmin)*(1-(gen-1)/Ngen)^bet;
end
endfunction
/////////MAIN ////////////////////////
Npop=300;Ngen=40;
n=2;
pc=0.2;pm=0.3;
xmin=1;xmax=20;
```

```
X=xmin+(xmax-xmin)*rand(Npop,n+1);
bestX=[];
for gen=1:Ngen
X=evaluation(X);   // evaluation
X=selection(X);    // selection
bestX=[bestX;best(X)];
Xnew=[];
for j=1:Npop/2
   u1=int(Npop*rand())+1;
   u2=int(Npop*rand())+1;
   if (rand()<pc) then
     Xnew=[Xnew;croisement(X([u1,u2],:))]; //croisement
   else
     Xnew=[Xnew;X([u1,u2],:)];
   end
end
for j=1:Npop
  if (rand()<pm) then
    Xnew(j,:)=mutation(Xnew(j,:),gen,Ngen,xmin,xmax);  // mutation
  end
end
X=Xnew;
end
//////////TRACE ////////////////////////
function  area= can(x)
d=x(1);h=x(2);
area=%pi*d^2/4+%pi*d*h
endfunction
////////////////////////////////
xset('window',0);clf
plot2d(Npop*(1:Ngen),bestX(:,$))
disp('meilleur individu (diameter,height):');disp(bestX($,1:n));
disp('volume:');disp(%pi*bestX($,1)^2*bestX($,2)/4);
disp('surface:');disp(can(X($,1:n)));
Xoptim=[(2400/%pi)^(1/3) ,1200/(%pi*(2400/%pi)^(2/3))];
disp('solution exacte:');disp(Xoptim);
////////////////////////////////////////
xset('window',1);clf;
xmin=1;xmax=12;ymin=1;ymax=12;
xcan=xmin:((xmax-xmin)/100):xmax;
ycan=ymin:((ymax-ymin)/100):ymax;
for i=1:101;
  for j=1:101;
    zcan(i,j)=can([xcan(i),ycan(j)]);
  end
  hcan(i)=300*4/(%pi*xcan(i)^2);
end
plot2d(bestX(:,1),bestX(:,2),rect=[xmin,ymin,xmax,ymax]);
plot2d(bestX($,1),bestX($,2),-2);
plot2d(Xoptim(1),Xoptim(2),-3);
plot2d(xcan,hcan,2,rect=[xmin,ymin,xmax,ymax]);
contour2d(xcan,ycan,zcan,10,rect=[xmin,ymin,xmax,ymax]);
```