TP 5 Examen

2 mai 2012

Commencez par lire attentivement tout l'énoncé. Vous pouvez écrire les réponses directement sur l'énoncé. Soignez votre écriture et utilisez un brouillon. N'oubliez pas d'écrire vos nom, prénom et numéro d'étudiant.

Nom :
Prénom :
N° Etudiant :

Le tp noté comporte deux parties séparées. La partie 1 consiste en quelles manipulations de commandes **scilab**. La partie 2, plus longue, concerne l'optimisation linéaire.

1 Commandes Scilab

Exercice 1 (Un programme non optimisé)
Dans ce premier exercice, il s'agit d'étudier une fonction **scilab** dont le corps est le suivant :

```
function bool=PROGScilab(A)
    bool=0;
    test=0;
    s=size(A);
    dif=(s(1)-s(2))*(s(1)-s(2));
    if(dif == 0)
        for i=1:s(1)
            for j=1:s(2)
                if((i-j)*(i-j)>0)
                    if(A(i,j)== -A(i,j))
                        test=test+1;
                    end
                end
            end
        end
    end
        if(test==s(1)*(s(2)-1))
            bool=1;
        end
        return bool;
endfunction
```

1. Que dire sur la matrice `A` si la variable `dif` est strictement positive ?

2. Quelles sont les valeurs possibles retournées par la fonction `PROGScilab` ?

3. Que teste la fonction `PROGScilab` ?

Exercice 2 (Construction d'un code réduit)
Dans tout l'exercice toutes les matrices sont supposées de taille $n \times m$ (n lignes et m colonnes). Les variables i et j représentent les indices correspondant respectivement à la ligne i et à la colonne j, ainsi pour une matrice M, $M(i,j)$ désigne le coefficient de M associé à la ligne i et à la colonne j.

Test d'égalité entre matrices. La commande `C=(A==B)` de **scilab** permet construire la matrice `C` composée de valeur booléenne (vrai ou faux comme dans un tableau de vérité). La matrice est construite de la manière suivante, pour tout i, j tels que $1 \leq i \leq n$, $1 \leq j \leq m$:

$$\begin{cases} \texttt{C(i,j)} = \texttt{T} & \text{si } \texttt{A(i,j)} = \texttt{B(i,j)} \\ \texttt{C(i,j)} = \texttt{F} & \text{si } \texttt{A(i,j)} \neq \texttt{B(i,j)} \end{cases}$$

Les variables `T` et `F` représentent les *variables booléennes* vrai et faux. Dans ce cas, le coefficient booléen retourné est vrai dès que les coefficients `A(i,j)` et `B(i,j)` sont égaux et faux sinon. Les commandes `%T` et `%F` permettent de rentrer sous **scilab** respectivement les variables booléennes `T` et `F`.

Conversion booléen/matrices 0-1. La commande `D=bool2s(C)` convertit une matrice à coefficients booléens de la manière suivante pour $1 \leq i \leq n$, $1 \leq j \leq m$:

$$\begin{cases} \texttt{D(i,j)} = 1 & \text{si } \texttt{C(i,j)} = \texttt{T} \\ \texttt{D(i,j)} = 0 & \text{si } \texttt{C(i,j)} = \texttt{F} \end{cases}$$

Minimum des coefficients d'une matrice. La commande `min(A)` retourne la plus petite valeur des coefficients de la matrice `A`. Par exemple, lorsque :

$$\texttt{A} = \begin{pmatrix} 1 & 0 \\ 2 & 3 \end{pmatrix}$$

la commande `min(A)` retourne `0`.

1. Que vaut la matrice `C` définie par `C=(A==B)` si

$$M = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 11 & 12 & 13 & 14 & 15 & 16 & 17 & 18 \\ 21 & 22 & 23 & 24 & 25 & 26 & 27 & 28 \\ 10 & 15 & 25 & 35 & 40 & 45 & 50 & 55 \end{pmatrix}, \quad \texttt{A} = M * M^T, \quad \texttt{B} = (\texttt{A} + \texttt{A'})/2 \ ?$$

Pour rappel, la commande `A'` désigne la transposée de la matrice `A`. Et la notation M^T désigne également la transposée de M (sans considérer l'implémentation **scilab**).

2. Que vaut la matrice `D` définie par `D=bool2s(C)` si

$$\texttt{C} = \begin{pmatrix} T & F & F \\ F & T & F \\ F & F & T \end{pmatrix} \ ?$$

3. Soit une matrice `A`. On suppose que pour tout i, j, $1 \leq i \leq n$, $1 \leq j \leq m$, `A(i,j)` $\in \{0, 1\}$ (`A(i,j)` vaut soit `0` soit `1`). Que retourne `min(A)` :
   (a) s'il existe i, j tels que `A(i,j)` $= 0$?
   (b) Si pour tout i, j, `A(i,j)` $= 0$?
   (c) Si pour tout i, j, `A(i,j)` $= 1$?

4. Compléter grâce à la commande `diag` de **scilab** et des commandes précédentes, la deuxième ligne de la fonction `PROGScilab2` suivante pour que celle-ci retourne les mêmes valeurs que `PROGScilab` :

```
function b=PROGScilab2(A)
    b=
    return b;
endfunction
```

Remarque 1
Les fonctions `PROGScilab` et `PROGScilab2` retournent les mêmes valeurs si pour toute matrice `A`, si `b=PROGScilab(A)` et si `c=PROGScilab2(A)` alors `b=c` .

2 Optimisation linéaire

On vous redonne le fonction de pivot de Gauss-Jordan à utiliser pour les problèmes d'optimisation linéaire. La fonction `PIVOTGJ` permet de faire rentrer la m-ième variable (colonne de la matrice `M`) en tant que j-ième élément de la base (ou de manière équivalente de faire sortir la j-ième variable de la base courante).

```
function N=PIVOTGJ(M,m,j)
    s=size(M);
    for i=1:s(1)
        if(i==j)
            M(i,:)=M(i,:)/M(i,m);
        else
            M(i,:)=M(i,:)-(M(i,m)/M(j,m))*M(j,:);
        end
    end
    N=M;
endfunction
```

Rappel :

1. Le **critère de Dantzig**
   (a) **Variable entrante :** La variable entrante est la variable dont le coût marginal est le plus grand ;
   (b) **Variable sortante :** La variable sortante est celle dont le rapport entre le second membre des contraintes et le coefficient de la matrice des contraintes est minimal. Ces rapports sont évalués uniquement sur la colonne sélectionnée pour la variable entrante et uniquement si les coefficients de la matrice sont strictement positifs. Si pour la colonne sélectionnée, tous les coefficients de la matrice sont négatifs ou nuls, le problème n'est pas borné.

   Dans le cas où deux indices satisfont le critère de Dantzig, l'indice le plus petit sera sélectionné.
2. Le **critère de Bland** consiste à seléctionner le plus petit indice des coûts marginaux strictement positifs comme indice de variable entrante. Le choix de la variable sortante est identique à celui du critère de Dantzig.

Exercice 3 (Premier problème)
On s'intéresse au problème de programmation linéaire dont la forme initiale est la suivante :

$$\begin{gathered}\text{Maximiser } Z(x,y,z) = x+y+z,\\ \text{sous les contraintes}\\ \begin{cases} x & & & & & \leq & 10\\ -3x & - & \frac{1}{4}y & + & z & \leq & 4\\ & & y & & & \leq & 20\\ & & & & z & \leq & 40\\ & & & & x,y,z & \geq & 0. \end{cases}\end{gathered} \tag{1}$$

1. Questions préliminaires :
   (a) De quel type de problème s'agit-il ?

(b) Montrer que le problème (1) est réalisable et admet une solution optimale. On appelera C le polyèdre (convexe fermé) des contraintes.

(c) Mettre le problème (1) sous forme matricielle standard $AX = d$ en introduisant les variables d'écart e_1, e_2, e_3, e_4. L'indice des variables d'écart correspond à la ligne du système des contraintes associée.

(d) Ecrire le tableau final correspondant à cette dernière reformulation du problème (1) en écrivant le (ou les) vecteur(s) de coûts à la (aux) dernière(s) ligne(s) (avec éventuellement actualisation des coûts).

2. Résolution du problème :

(a) Donner une première solution de base réalisable.

(b) Utiliser 3 pivots de Gauss-Jordan (combiné avec le critère de Dantzig) pour résoudre le problème (1). Ecrire les appels successifs à la fonction `PIVOTGJ`, puis une (la) solution optimale ainsi que la valeur optimale.

(c) Combien de solutions possède le problème (1) ? Expliquer.

Exercice 4 (Deuxième problème)
On s'intéresse au problème de programmation linéaire dont la forme initiale est la suivante :

$$\text{Maximiser } Z(x,y,z) = 4x - 2y + z,$$
$$\text{sous les contraintes}$$
$$\begin{cases} 5x + y + 4z \leq 50 \\ y + z \geq 10 \\ x - z \geq 0 \\ x, y, z \geq 0. \end{cases} \tag{2}$$

1. Questions préliminaires :
   (a) De quel type de problème s'agit-il ?
   (b) Montrer que ce problème (2) est réalisable et admet une solution optimale. On appelera C le polyèdre (convexe fermé) des contraintes.
   (c) Mettre le problème (2) sous forme matricielle standard $AX = d$ en introduisant les variables d'écart et de faux-écart e_1, f_2, e_3. L'indice des variables d'écart correspond à la ligne du système des contraintes associée.
   (d) Une autre variable est-elle nécessaire ? Expliquer.
   (e) Ecrire le tableau final correspondant à cette dernière reformulation du problème (2) en écrivant le (ou les) vecteur(s) de coûts à la (aux) dernière(s) ligne(s) (avec éventuellement actualisation des coûts).
2. Résolution du problème :
   (a) Donner une première solution de base réalisable pour le problème (2).
   (b) Utiliser 2 pivots de Gauss-Jordan (combiné avec le critère de Dantzig) pour résoudre le problème (2).

(c) Combien de solutions possède le problème (2) ? Expliquer.

Exercice 5 (Problèmes de deuxième espèce)

On s'intéresse au problème de programmation linéaire dont la forme initiale est la suivante :

$$\begin{array}{c}\text{Maximiser } Z(x,y,z) = -x + y - 2z,\\ \text{sous les contraintes}\\ \left\{\begin{array}{rcrcrcl} x & + & 2y & - & z & \geq & 2\\ & & y & - & 2z & \geq & 5\\ x & & & + & z & \leq & 10\\ & & & & x,y,z & \geq & 0. \end{array}\right.\end{array} \tag{3}$$

1. Prouver que le problème (3) est réalisable (donner une solution réalisable).
2. Résoudre le problème (3). Combien de solutions possède le problème (3) ? Expliquer.

On s'intéresse au problème de programmation linéaire dont la forme initiale est la suivante :

$$\begin{array}{c}\text{Minimiser } Z(x,y,z) = -x + y - 2z,\\ \text{sous les contraintes}\\ \left\{\begin{array}{rcrcrcl} x & + & 2y & - & z & \geq & 2\\ & & y & - & 2z & \geq & 5\\ x & & & + & z & \leq & 10\\ & & & & x,y,z & \geq & 0. \end{array}\right.\end{array} \tag{4}$$

On admet que le problème (4) possède une solution optimale.

1. Mettre le problème (4) sous forme matricielle standard $AX = d$ en introduisant les variables d'écart et de faux-écart f_1, f_2, e_3. L'indice des variables d'écart correspond à la ligne du système des contraintes associée.

2. Ecrire le tableau final correspondant à cette dernière reformulation du problème (4) en écrivant le (ou les) vecteur(s) de coûts à la (aux) dernière(s) ligne(s) (avec éventuellement actualisation des coûts).

3. Donner une première solution de base réalisable pour le problème (4).

4. Résoudre le problème (4). Ecrire les appels successifs à la fonction `PIVOTGJ`, puis une (la) solution optimale ainsi que la valeur optimale. Combien de solutions possède le problème (4) ? Expliquer.

Exercice 6 (Modélisation : crème au chocolat blanc)
Un patissier débutant souhaite réaliser une crème au chocolat blanc. Son amie patissière confirmée lui indique qu'il a besoin des ingrédients suivants : chocolat blanc, jaunes d'oeuf, crème liquide entière et vanille (extrait d'une gousse).

Il doit réaliser **au moins 400 grammes de crème au chocolat blanc**. Son amie lui donne uniquement les proportions à respecter pour que sa crème soit réussie :

- la crème doit contenir **au moins 2.5 fois plus** de chocolat blanc que de crème liquide entière ;
- la crème doit contenir **au moins 2 fois plus** de crème liquide entière que de jaunes d'oeuf ;
- la crème doit contenir **au moins 4 fois plus** de jaunes d'oeuf que de vanille.
- la crème doit contenir **au moins 15 grammes** de vanille.

Cependant le patissier est débutant et souhaite réaliser sa crème **à moindre coût**. On introduit les variables x, y, z, t représentant la quantité (en grammes) des ingrédients. Le tableau suivant représente l'assignation des variables ainsi que le coût des ingrédients pour 100g (grammes) :

	Chocolat blanc	Jaune d'oeuf	Crème entière liquide	Vanille
Variable (en g)	x	y	z	t
Coût pour 100 g	2	1	1.5	30

1. Modéliser ce problème à l'aide d'un programme linéaire mis sous forme canonique (uniquement avec les variables x, y, z, t sans variables d'écart).

2. Trouver la quantité optimale de chaque ingrédient et indiquer le coût optimal de la crème au chocolat blanc.