IT45 TP 1
Modélisation mathématique

Ce TP est une introduction à l'optimisation par la modélisation mathématique des problèmes à traiter.

1 Présentation des outils

La résolution des problèmes de ce TP se fera par les outils suivants :
- logiciel de calcul scientifique GNU Octave : la fonction est « glpk » (commande « help glpk » pour plus d'informations)
- solveur AMPL
- solveur GLPK : 'glpsol' en ligne de commande, 'scite' sous linux, 'gusek' sous windows

2 Exercice : Assemblage de 2 produits

Une entreprise fabrique 2 produits P1 et P2 à partir de 3 composants a, b, c. Pour fabriquer un produit P1, l'entreprise a besoin de 2 composants a, 2 composants b et 1 composant c. Un produit P2 requiert, quant à lui, 3 composants a, 1 composant b et 3 composant c. Les disponibilités en composants a, b et c sont respectivement de 180, 120 et 150. Les gains marginaux des produits P1 et P2 sont de 3€ et 4€. On cherche à déterminer le nombre de produits P1 et P2 à fabriquer pour réaliser le bénéfice maximal.

1. En posant $x1$ et $x2$ le nombre de produits P1 et P2 fabriqués, formuler ce programme linéaire en indiquant la fonction objectif et les trois contraintes qui correspondent au nombre maximum de composants a, b et c disponibles.
2. Résoudre ce problème avec le logiciel de calcul scientifique Octave
3. Ecrire le modèle de ce programme en utilisant le formalisme AMPL
   (a) Déclarer 2 variables $x1$ et $x2$
   (b) Ecrire les trois contraintes relatives à la quantité de composants disponibles
   (c) définir l'objectif $objz$ à maximiser
4. Quelle est la solution optimale de ce programme linéaire et le gain correspondant ?

3 Exercice : Assemblage de n produits

On veut maintenant généraliser ce premier modèle à la fabrication de n produits à partir de m composants. Les hypothèses suivantes sont retenues. Le produit j rapporte un gain marginal c_j ; le nombre de composants i nécessaires pour fabriquer le produit j est $a_{i,j}$ et le nombre maximum de composants i est de b_i. Dans ces conditions :

1. On posant x_j le nombre de produits j fabriqués, écrire le nouveau modèle mathématique en précisant la fonction objectif et les contraintes
   (a) définir la fonction objectif en fonction des x_j et des gains unitaires c_j pour $j = 1..n$
   (b) définir pour tout composant $i = 1..m$ la contrainte qui limite son utilisation à sa quantité disponible b_i
2. Ecrire le programme paramétrique GMPL en utilisant les données de l'exercice précédent pour alimenter le modèle.
   (a) déclaration des ensembles d'indices
   - déclarer un paramètre *nbr_prod* à 2 et *nbr_compo* à 3

- déclarer un ensemble de produits nommé $PRODUCTS$ de 1 à *nbr_prod* et un ensemble de composants $COMPO$ de 1 à *nbr_compo*

(b) Déclaration des paramètres et des variables
- déclarer les gains des produits comme un paramètre indicé sur l'ensemble des produits
- déclarer le tableau relatif au nombre maximum de composant $b$indicé sur les composants
- déclarer le tableau a, indicé sur les ensembles $COMPO$ et $PRODUCTS$. $a[i][j]$ représente le nombre de composants i pour fabriquer le produit j
- déclarer la variable x indicé sur les produits fabriqués

(c) Ecriture des contraintes et de l'objectif
- définir une contrainte indicée qui vérifie que pour chaque composant sa consommation est inférieure à sa disponibilité en stock
- définir l'objectif à maximiser par une somme indicée sur tous les produits du gain unitaire multiplié par la quantité fabriquée

(d) Initialisation des paramètres dans la section *data*
- initialiser le tableau c avec les valeurs 3 et 4
- faire de même pour les autres paramètres

4 Crème glacée

Une société désire produire 100 kg d'une préparation de base pour crème glacée. Cette préparation doit être constituée de 21.5% de matières grasses, 21% de sucre, 1.2% d'oeuf et 56.3% d'eau.

Pour réaliser cette préparation, la société dispose des ingrédients récapitulés dans la liste suivante, pour lesquels sont indiquées les proportions de matières grasses, de sucre, d'oeuf et d'eau (entre parenthèses), ainsi que le coût en euro au kg.
- crème (40% MG, 60% eau) : 0.45 euro/kg.
- jaune d'oeuf frais (50% MG, 40% oeuf, 10% eau) : 0.6 euro/kg.
- lait entier en poudre (12% MG, 88% eau) : 0.15 euro/kg.
- jaune d'oeuf surgelé, sucré (30% MG, 14% sucre, 40% oeuf, 16% eau) : 0.3 euro/kg.
- sirop de sucre de canne (70% sucre, 30% eau) : 0.12 euro/kg.
- eau (100% eau) : 0 euro/kg.

1. Modéliser ce problème sous forme de programme linéaire
2. Jusqu'à présent, le fabricant produisait le mélange suivant : 50 kg de crème, 3 kg de jaune d'oeuf frais, 30 kg de sirop et 17 kg d'eau. Vérifier que ce mélange vérifie les contraintes et donner son coût.
3. Vérifier la composition du mélange de coût minimal avec un logiciel de calcul scientifique
4. Ecrire le programme paramétrique GMPL et résoudre le problème avec *GLPK*.
5. Les prix de la crème et du jaune d'oeuf frais passent respectivement à 0.6 et 1 euro. Quelle est la nouvelle composition de coût minimal ?

5 Choix de sites pour des entrepôts

Une entreprise de distribution envisage l'implantation de m nouveaux entrepôts. n sites ($n > m$) ont été envisagés pour leur implantation et, pour chaque site si, on a estimé la capacité de stockage c_i de l'entrepôt que l'on pourrait construire sur ce site.

L'entreprise cherche à déterminer les m sites sur lesquels elle doit construire ses nouveaux entrepôts de façon à maximiser la capacité totale de stockage. Mais elle désire éviter à tout prix d'avoir 2 entrepôts situés à plus de 30 kms l'un de l'autre car elle effectue de très nombreux transports entre tous ses entrepôts. (Les m sites qui ont été envisagés sont toujours situés à moins de 30 kms de tous les entrepôts déjà construits).

1. Ecrire le modèle mathématique de ce problème en introduisant :
   (a) une variable x_i qui vaut 1 si le site i est sélectionné pour y construire un entrepôt
   (b) un paramètre δ_{ij} qui vaudra 1 si les sites i et j sont distants de plus de 30 kms.

2. Résoudre le problème d'implantation à $m = 3$ entrepôts et $n = 6$ sites dont les capacités sont les suivantes : 134, 167, 189, 182, 136 et 192. Les distances entre sites sont données dans la matrice suivante :

$$dist = \begin{array}{|c|cccccc|} \hline & 1 & 2 & 3 & 4 & 5 & 6 \\ \hline 1 & \infty & 55 & 52 & 20 & 24 & 24 \\ 2 & 55 & \infty & 43 & 40 & 43 & 36 \\ 3 & 52 & 43 & \infty & 53 & 34 & 27 \\ 4 & 20 & 40 & 53 & \infty & 51 & 25 \\ 5 & 24 & 43 & 34 & 51 & \infty & 17 \\ 6 & 24 & 36 & 27 & 25 & 17 & \infty \\ \hline \end{array}$$

3. Modifier le modèle pour que la distance de 30 kms (au delà de laquelle 2 sites ne peuvent pas être retenus) devienne un paramètre de l'instance. Pour cela, on génèrera directement les coefficients δ_{ij} à partir de la matrice des distances avec une expression indicée de la forme :

```
param delta{ i in SITES, j in SITES} := if condition then valeur1 else valeur2 ;
```

6 Annexes

6.1 Solveur AMPL

Pour résoudre un problème :

1. Réinitialiser le modèle avec la commande :

```
reset;
```

2. Sélectionner un solver

```
option solver highs;
option solver cbc;
option solver scip;
...
```

3. Définir le modèle et les données

```
model [FILENAME]
data [FILENAME]
```

4. Résolution

```
solve;
```

5. Affichage solutions

```
display solve_result_num, solve_result;
display VAR;
```

6.2 Solveur GLPK

La documentation de GLPK est accessible dans : « /usr/share/doc/glpk-doc/ »
Pour résoudre un problème nommé « pb.mod » au format GMPL, deux possibilités :

1. En ligne de commande :

```
glpsol --math pb.mod --output res.log
```

2. En utilisant l'éditeur de texte *SciTE*(ou 'gusek' sous windows) :

   (a) la résolution se fait par l'option « **Tools/Go** (**F5**) »

   (b) (**F1**) affiche la doc. de GMPL (GNU MathProg Language)

6.3 Résolution d'un programme linéaire avec Octave

Pour résoudre un programme linéaire, voir la documentation `https://docs.octave.org/latest/Linear-Programming`

6.4 Résolution d'un programme linéaire avec Matlab

Soit le programme linéaire à modéliser avec Matlab :

$$\begin{cases} max\, z = & 5x_1 & +7x_2 & & & \\ & 3x_1 & +5x_2 & = & 100 & (1) \\ & x_1 & +2x_2 & \leq & 200 & (2) \\ & & x_i & \geq & 0 & \end{cases}$$

Déclaration de variables X1 et X2 positives :

```
X1 = optimvar(>X1','LowerBound',0);
X2 = optimvar('X2','LowerBound',0);
```

Création du problème et de l'objectif (Matlab minimise l'objectif) :

```
linprob = optimproblem('Objective',-5*X1-7*X2);
```

Définition des contraintes :

```
linprob.Constraints.cons1 = 3*X1 + 5*X2 == 100 ;
linprob.Constraints.cons2 = X1 + 2*X2 <= 200 ;
```

Résolution du programme linéaire :

```
linsol = solve(linprob);
```

Affichage de la valeur de la solution :

```
evaluate(linprob.Objective,linsol)
```

Affichage de la solution :

```
tbl = struct2table(linsol)
```

Aide en ligne de Matlab : `https://fr.mathworks.com/help/optim/ug/example-linear-programming-via-problem.html`