

Programmation stochastique

SMPS

Fabian Bastin

IFT-6512 – Hiver 2011

Formulation "par colonnes" du problème, avec les sections suivantes :

- NAME
- ROWS
- COLUMNS
- RHS
- RANGES
- BOUNDS
- ENDATA

Format fixe ! Les champs commencent dans les colonnes 1, 5, 15, 25, 40 et 50 (souvenir de l'époque des cartes perforées).

L'**ordre des sections** doit aussi être respecté.

Un exemple : ouch !

(Honteusement) repris de Wikipédia. . .

Supposons que nous souhaitions résoudre le problème

$$\min x + 4y + 9z$$

$$\text{t.q. } x + y \leq 5$$

$$x + z \geq 10$$

$$-y + z = 7$$

$$0 \leq x \leq 4$$

$$-1 \leq y \leq 1$$

Ouch!

```
NAME                TESTPROB
ROWS
  N  COST
  L  LIM1
  G  LIM2
  E  MYEQN
COLUMNS
  XONE      COST      1    LIM1      1
  XONE      LIM2      1
  YTW0      COST      4    LIM1      1
  YTW0      MYEQN     -1
  ZTHREE    COST      9    LIM2      1
  ZTHREE    MYEQN     1
RHS
  RHS1      LIM1      5    LIM2      10
  RHS1      MYEQN     7
BOUNDS
  UP BND1   XONE      4
  LO BND1   YTW0     -1
  UP BND1   YTW0      1
ENDATA
```

- Et une instance de programmation stochastique ?
- Nous pourrions former nous-mêmes l'équivalent déterministes. . . Vous en avez vraiment envie ?
 - Pour des problèmes vraiment gros, former l'équivalent déterministe est hors de question.
 - Nous devons juste spécifier les parties aléatoires du modèle.
- Nous pouvons faire ceci en utilisant le format SMPS.
- Il y a des efforts pour traiter les problèmes stochastiques avec des langages de modélisation algébrique, mais ce n'est pas encore parfaitement au point.
Au niveau de COIN-OR, pour les problèmes linéaires stochastiques, il y a des efforts d'intégration en FLOPC++ et Smi.

Composantes d'un fichier SMPS

En fait, je devrais dire des fichiers. 3 fichiers seront présents pour décrire un problème :

- fichier "**core**" : semblable à un fichier MPS, pour l'instance de base ;
- fichier "**time**" : spécifie la structure de dépendance temporelle ;
- fichier "**stoch**" : spécifie l'aléatoire.

Le site **à visiter** pour mieux comprendre ce format :

<http://myweb.dal.ca/gassmann/smeps2.htm>

Référence : Gassmann et Kristjánsson, *The SMPS format explained*, IMA Journal of Management Mathematics, 2007, <http://imaman.oxfordjournals.org/cgi/reprint/dpm007v1>

Plusieurs exemples sont disponibles dans le sous-répertoire Data/Stochastic de Smi.

Hypothèse : on regarde les périodes comme couple "événement-décision" (dans cet ordre).

Smi est capable de lire les fichiers SMPS et de les passer aux solveurs linéaires respectant l'interface OSI définie dans COIN-OR.

Solveur linéaire de COIN : CLP (Coin-OR Linear Programming) :

<https://projects.coin-or.org/Clp>

L'interface OSI et le solveur Clp sont inclus avec Smi.

Inteface OSI :

`Osi/src/OsiClp/OsiClpSolverInterface.hpp`

Source : Smi/examples/stoch.cpp

```
SmiScnModel smi;
```

```
smi.readSmps(name);
```

```
OsiClpSolverInterface *clp = new OsiClpSolverInterface();
```

```
smi.setOsiSolverHandle(*clp);
```

```
OsiSolverInterface *osiStoch = smi.loadOsiSolverData();
```

```
osiStoch->setHintParam(OsiDoPresolveInInitial, true);
```

```
osiStoch->setHintParam(OsiDoScale, true);
```

```
osiStoch->setHintParam(OsiDoCrash, true);
```

```
osiStoch->initialSolve();
```

Solution

```
printf("Solved_stochastic_program_%s\n", name);  
printf("Number_of_rows: %d\n", osiStoch->getNumRows());  
printf("Number_of_cols: %d\n", osiStoch->getNumCols());  
printf("Optimal_value: %g\n", osiStoch->getObjValue());
```

```
string outfilename(name);  
const string suffix(".out");  
outfilename = outfilename + suffix;  
FILE *fp = fopen(outfilename.c_str(), "w");  
int numScenarios=smi.getNumScenarios();
```

```
for (int i=0 ; i<numScenarios; ++i) {  
    double *dsoln=NULL;  
    int numCols=0;  
    fprintf(fp, "Scenario_%d_\n", i);  
    dsoln = smi.getColSolution(i, &numCols);  
    for (int j=0; j<numCols; j++)  
        fprintf(fp, "%g_\n", dsoln[j]);  
    free(dsoln);  
}  
fclose(fp);
```

But : définir le nom du problème ainsi que le type de chaque contrainte, les noms des lignes et des colonnes, une liste des coefficients de la matrice de contraintes par colonne, les coefficients du terme de droite et les bornes et domaines sur les variables du problèmes et les variables d'écart.

Cas de base : les éléments stochastiques doivent être mentionnés, avec une valeur préliminaire (ne représentant pas nécessairement quelque chose de pertinent pour le problème considéré). On définit donc un problème **déterministe**.

Notation : les commentaires commencent par *, et la ligne suivante nous permet d'identifier le numéro de colonne.

```
*23456789 123456789 123456789
```

On ne reprendra ici que les principaux éléments.

Structure d'une ligne

Une ligne contient une entête ou une ligne de donnée.

Ligne de données :

- 1 colonne 2 : champ de code (C) ;
- 2 colonne 5 : premier champ de nom (N1), colonne 15 : deuxième champ de nom (N2) - 10 caractères chacun ;
- 3 colonne 25 : première valeur numérique ;
- 4 colonne 40 : second champ de nom (N2) - 10 caractères ;
- 5 colonne 50 : second champ numérique (V2) - 10 caractères ;

```
*23456789 123456789 123456789 123456789 123456789 123456789
  C  N1          N2          V1          N2          V2
```

Ligne d'entête :

```
*23456789 123456789 123456789 123456789 123456789 123456789
N1          N2          N3
```

Sections : NAME, ROWS

NAME : nom du problème (doit coïncider dans **time** et **stoch**).

```
*23456789 123456789 123456789  
NAME           Example
```

ROWS : lignes du problème. Le type est donné par une lettre dans le champ approprié :

- L inférieur ou égal ;
- G supérieur ou égal ;
- E égal ;
- N ligne de fonction (pas de contrainte), en particulier l'objectif. Par défaut, la première ligne de type 'N' est utilisée comme objectif.

Chaque ligne reçoit un identifiant (défini par l'utilisateur) dans le premier champ de noms.

Sections : ROWS, COLUMNS

```
*23456789 123456789 123456789
```

```
ROWS
```

```
  N  OBJECT
```

```
  G  ROW2
```

COLUMNS : reprend les coefficients non nuls de la matrice de contraintes, par colonne (souvenir de `Fortran...`). Le premier champ nom contient la colonne, le second, la ligne et le premier champ numérique contient la valeur du coefficient. Une seconde valeur peut être entrée dans le second champ numérique, en utilisant le second champ de nom.

```
*23456789 123456789 123456789 123456789 123456789 123456789
```

```
COLS
```

```
  COL1
```

```
  ROW2
```

```
  2.0
```

```
  OBJECT
```

```
  100.
```

RHS

- Premier champ nom : identifiant.
- La ligne (i.e. la contrainte) est indiquée dans le second champ de nom.
- Le premier champ numérique contient la valeur.
- On peut indiquer une seconde valeur en utilisant le troisième champ de nom et le second champ numérique.
- On ne doit pas citer les valeurs nulles.

```
*23456789 123456789 123456789
```

```
RHS
```

```
    RHS1      ROW2      50.0
```

Sections : RANGES

RANGES permet de définir des contraintes d'égalité \mathbb{L} et \mathbb{G} (i.e. définir des bornes sur une combinaison linéaire des variables). Le premier champ nom donne un identifiant, le second la contrainte, et le troisième une valeur r_i modifiant la borne :

G	$b_i \leq \sum a_j x_j \leq b_i + r_i $
L	$b_i - r_i \leq \sum a_j x_j \leq b_i$
E, $r_i > 0$	$b_i \leq \sum a_j x_j \leq b_i + r_i$
E, $r_i < 0$	$b_i + r_i \leq \sum a_j x_j \leq b_i$

```
*23456789 123456789 123456789
```

```
RANGES
```

```
    RANGE1
```

```
    ROW2
```

```
    10.0
```

Sections : BOUNDS

- Champ de code : type de contrainte :
 - LO : borne inférieure, UP : borne supérieure ;
 - FX : valeur fixe, FR : variable libre ;
 - MI : variable non-positive ; PL : variable non-négative ;
 - BV : variable binaire (0 ou 1).
 - UI : borne supérieure, arrondie au plus proche entier si la variable est entière.
- Premier champ de nom : identifiant ; deuxième champ de nom : nom de la variable.
- Champ de valeur : valeur de la borne.

Les bornes par défaut sont 0 et $+\infty$ pour des variables continues et 0 et 1 pour des variables entières.

```
*23456789 123456789 123456789
BOUNDS
UP BOUND1      COL1      20.
```

Des variables entières peuvent être définies à l'aide de champs **indicateurs**, insérés avant le début et à la fin d'un groupe de variables entières.

Syntaxe : INT dans le premier champ de nom, 'MARKER' dans le second champ de nom et 'INTORG' ou 'INTEND' dans le troisième champ, pour indiquer respectivement le début ou la fin du groupe de variables.

```
*23456789 123456789 123456789 123456789 123456789 123456789
  INT          'MARKER'
  Col1         Row1          1.0
  Col2         Row1         -1.0
  INT          'MARKER'
                'INTEND'
```

Objectif quadratique

Peut être défini à la suite de la partie linéaire. 3 parties :

NAME nom, devant correspondre avec la ligne concernée, définie dans la partie linéaire ;

QSECTION permet de spécifier les coefficients non nuls ;

ENDATA fin des données du problème.

Exemple : pour définir l'objectif, avec un facteur commun 0.5,

$$0.5(\text{Col1})^2 + 1.5\text{Col1Col2} + (\text{Col2})^2.$$

```
*23456789 123456789 123456789 123456789
NAME          Example
QSECTION
    Col1      Col1      1.0
    Col1      Col2      3.0
    Col2      Col2      2.0
```

Le fichier **time** sert à séparer le fichier **core** en séparant le scénario de base correspondant aux étapes individuelles.

TIME donne un nom au problème dans le second champ de nom.

PERIODS donne les noms de chaque période de temps (dans l'ordre).

ROWS spécifie pour chaque ligne la période à laquelle elle appartient.

COLUMNS spécifie pour chaque colonne la période à laquelle elle appartient.

ENDATA fin des données du problème.

```
*23456789 123456789 123456789  
TIME          EXAMPLE
```

Section PERIODS

Entête : PERIODS suivi d'IMPLICIT si le fichier `core` est ordonné suivant le temps. Le second champ de nom peut aussi être laissé blanc dans ce cas.

Sinon, il convient d'employer le mot-clé `EXPLICIT` et de faire suivre les sections `ROWS` et `COLUMNS`.

Chaque enregistrement de donnée contient le nom de la période dans le troisième champ de nom. Si le format est explicite, les deux premiers champs sont inusités, sinon, ils contiennent l'indicatif de colonne et de nom. La ligne d'objectif doit toujours appartenir à la première période.

```
*23456789 123456789 123456789 123456789
PERIODS          IMPLICIT
      COL1      ROW1          PER1
      COL5      ROW3          PER2
```

Section ROWS, COLUMNS

La section **ROWS** contient le nom d'une ligne mentionnée dans le fichier **core** dans le premier champ de nom et le nom de la période dans le second champ de nom.

```
*23456789 123456789 123456789
```

```
ROWS
```

```
    ROW1          PER1
```

```
    ROW3          PER2
```

```
    ROW2          PER1
```

La section **COLUMNS** fonctionne de même, mais pour les colonnes.

```
*23456789 123456789 123456789
```

```
COLS
```

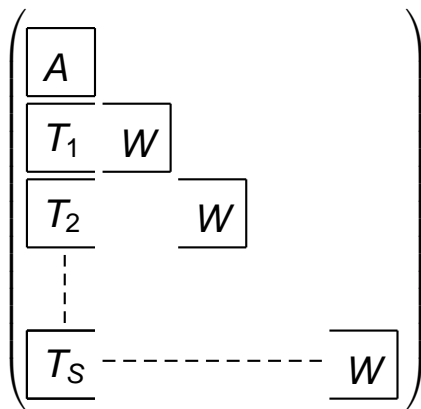
```
    COL1          PER1
```

```
    COL5          PER2
```

```
    COL2          PER1
```

Affectation aux différentes étapes

Motivation : structure d'un programme linéaire à 2 étapes.

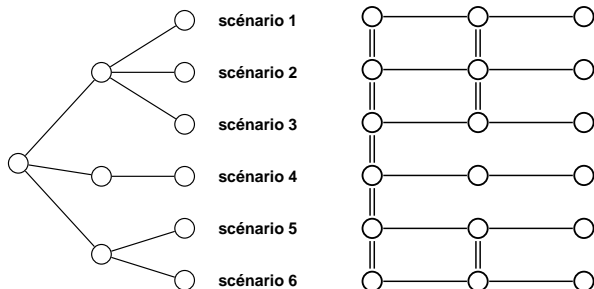


Affectation aux différentes étapes (2)

- 1 Une colonne appartient à l'étape où la décision est prise.
- 2 Une ligne appartient à l'étape de la dernière décision afférente à la ligne.
- 3 Une variable aléatoire appartient à la première étape ayant un élément stochastique dépendant de cette variable.
- 4 Pour les distributions finies, toutes les colonnes (et avec elles les lignes) doivent être répliquées suivant les variables aléatoires de même étape.

But : permettre la construction de l'équivalent déterministe.

Si toutes les variables aléatoires sont de support fini, cela revient à la construction d'un arbre de scénarios, qui peut être décrit de **3 manières différentes** : scénario par scénario, noeud par noeud, ou implicitement en utilisant des distributions marginales.



Fichier stoch : sections

- STOCH** nom du problème ;
- SIMPLE** (optionnel) problèmes avec recours simple ;
- ROBUST** (optionnel) pénalités quadratiques ;
- PLINQUAD** (optionnel) pénalités quadratique-linéaire par morceaux ;
- CHANCE** (optionnel) contraintes en probabilité ;
- ICC** contraintes en probabilité intégrées ;
- SCENARIOS** valeurs des éléments stochastiques, un scénario à la fois ;
- NODES** valeurs des éléments stochastiques, un noeud à la fois ;
- INDEP** distribution marginale de variables aléatoires indépendantes ;
- BLOCKS** distribution marginale d'un vecteur aléatoire ;
- DISTRIB** distribution de variables aléatoires auxiliaires ;
- ENDATA** fin des données du problème.

STOCH

*23456789 123456789 123456789

STOCH

Example

SIMPLE : recours simple. On pénalise les écarts à la relation

$$\sum a_{ij}x_j = b_i,$$

de la manière suivante :

$$q_i^- b_i^- = q_i^- (b_i - \sum a_{ij}x_j), q_i^+ b_i^+ = q_i^+ (\sum a_{ij}x_j - b_i).$$

Chaque enregistrement de donnée dans cette section contient le nom d'une ligne, suivie des coefficients q^+ et q^- , avec la condition

$$q^+ + q^- \geq 0.$$

(Exercice : pourquoi ?) Cette contrainte est forcée par système si elle est violée.

Section SIMPLE, SCENARIO

SIMPLE : chaque occurrence d'un nom de ligne dans cette section déclenche la génération de variables de décision additionnelles, identifiées en ajoutant le suffixe `.minus` et `.plus` au nom de la ligne. Pour une contrainte d'inégalité, seule la variable d'écart susceptible d'être utilisée est générée.

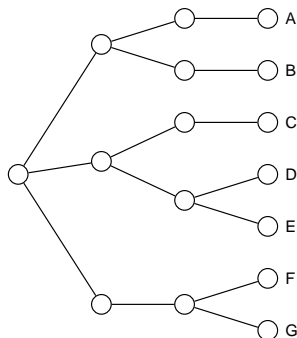
```
*23456789 123456789 123456789 123456789 123456789 123456789
SIMPLE
      SIMPLE1      ROW1          10.0          5.0
      SIMPLE1      ROW2          50.0
```

SCENARIOS : spécifie l'arbre de scénarios (ô surprise !). Pour garder le fichier condensé, on ne spécifie un nouveau scénario qu'au point où il se sépare (branche) d'un scénario déjà défini.

Section SCENARIO

Le début d'un scénario est donné par SC, suivi du nom du scénario, et du noeud à partir duquel il branche. Le champ numérique donne la probabilité de ce scénario, et le troisième champ indique la période de branchement.

Les champs de données sont les valeurs des variables pour ce scénario, en utilisant les noms définis dans le fichier [core](#).



Section SCENARIO (2)

```
*23456789 123456789 123456789 123456789 123456789
SCENARIOS
SC SCEN_A      'ROOT'      0.3          PERIOD1
      COL2      ROW3          1.0
      COL3      ROW4          1.0
UP BOUND1      COL5 1.0
SC SCEN_B      SCEN_A      0.2          PERIOD3
      COL3      ROW4          2.0
UP BOUND1      COL5          2.0
SC SCEN_C      SCEN_A      0.1          PERIOD2
      COL2      ROW3          2.0
      COL3      ROW4          2.0
UP BOUND1      COL5          2.0
SC SCEN_D      SCEN_C      0.1          PERIOD3
      COL3      ROW4          1.0
UP BOUND1      COL5          1.0
SC SCEN_E      SCEN_D      0.1          PERIOD4
UP BOUND1      COL5          2.0
```

Section SCENARIO, NODE

```
SC SCEN_F      SCEN_A      0.1          PERIOD2
   COL2        ROW3        3.0
   COL3        ROW4        3.0
UP BOUND1      COL5        3.0
SC SCEN_G      SCEN_F      0.1          PERIOD4
   COL4        ROW5        2.0
```

La section **NODE** permet de spécifier l'arbre de scénarios noeud par noeud, ce qui est pratique notamment quand la profondeur de chaque branche est différente, ou quand le nombre de lignes ou de colonnes associé à une période particulière dépend de l'historique du processus.

Ce ne sera pas le cas dans nos exemples. Cette description est beaucoup plus complexe à manipuler.

Sert à modéliser les variables aléatoires. 3 possibilités :

- 1 distribution discrète ;
- 2 distribution spéciale avec au plus 2 paramètres ;
- 3 routine utilisateur.

Le champ d'entête distingue entre ces formes en plaçant le mot-clé approprié dans le second champ de nom. La valeur de la variable aléatoire remplace la valeur du cas de base, mais ce comportement peut être modifié en remplaçant le troisième champ de nom par la valeur `ADD`, `MULTIPLY`, `REPLACE`.

Les enregistrements de données reprennent la position (colonne, ligne), le premier paramètre numérique, la période (qui peut être laissée vide si elle peut être inférée de la position), le second paramètre numérique.

Section INDEP (2)

Les distributions supportées directement sont DISCRETE, UNIFORM, NORMAL, GAMMA, BETA, LOGNORM. Voir Gassmann !

```
*23456789 123456789 123456789 123456789 123456789 123456789
INDEP          DISCRETE
      COL1      ROW8          6.0          PERIOD2          0.5
      COL1      ROW8          8.0          PERIOD2          0.5
      RHS       ROW8          1.0          PERIOD2          0.1
      RHS       ROW8          2.0          PERIOD2          0.5
      RHS       ROW8          3.0          PERIOD2          0.4
```

Définition une distribution triangulaire entre 10 et 20, et de mode donné par [ROW1 , COL1], pour répondre à une sous-routine TRIANG(min, max, mode)

```
*23456789 123456789 123456789 123456789 123456789 123456789
INDEP          TRIANG
      COL5      ROW8          PERIOD2
PR
          10.0
          20.0
      COL1      ROW1
```

BLOCKS permet de définir des vecteurs aléatoires dont les composantes sont corrélées, mais indépendants des autres vecteurs du problème.

D'autres sections et possibilités sont décrites dans le format SMPS. Nous ne les utiliserons pas ici, mais vous êtes invités à lire la documentation afférente pour connaître celles-ci.

A terme, il devrait être préférable de pouvoir utiliser le langage de modélisation associé à FlopC++ en combinaison avec Coin. Beaucoup plus souple !