Optimisation discrète sous incertitudes

Christian Artigues

LAAS-CNRS

22 septembre 2022

Optimisation

Définition

- X : ensemble de solutions
- x : un élément de l'ensemble
- $f : X \to \mathbb{R}$: fonction **objectif**

Problème d'optimisation: trouver $x^* = \text{argmin}_{x \in X} f(x)$

Optimisation discrète / continue

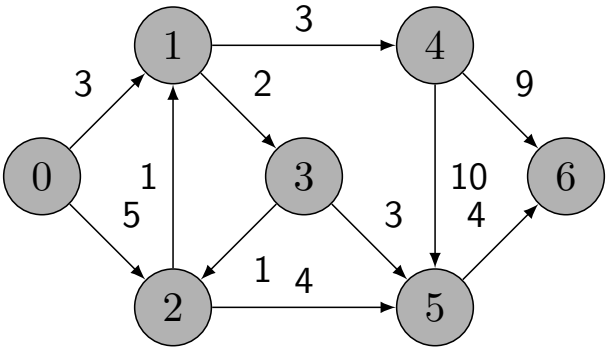
- X : ensemble discret de solutions $\implies$ Optimisation discrète
- $X = \{\mathbf{x} \in \mathbb{R}^n | g_j(\mathbf{x}) \geq 0, j \in 1, \ldots, m\}$, avec $n, m \in \mathbb{N}$ et g_j fonctions continues de $\mathbb{R}^n \to \mathbb{R}$ $\implies$ Optimisation continue

Exemple de problème d'optimisation déterministe

Plus court chemin dans un graphe orienté

Définitions

- V ensemble de $n+2$ sommets, où 0 et $n+1$ sont les sommets de début et de fin.
- E ensemble d'arcs tels que l_{ij} est la valeur de l'arc $(i,j) \in E$.
- Ensemble de solutions X : tous les chemins de 0 à $n+1$.
- Longueur d'un chemin : somme des valeurs des arcs du chemin.
- PCC : Trouver le chemin $x \in X$ de longueur minimale.



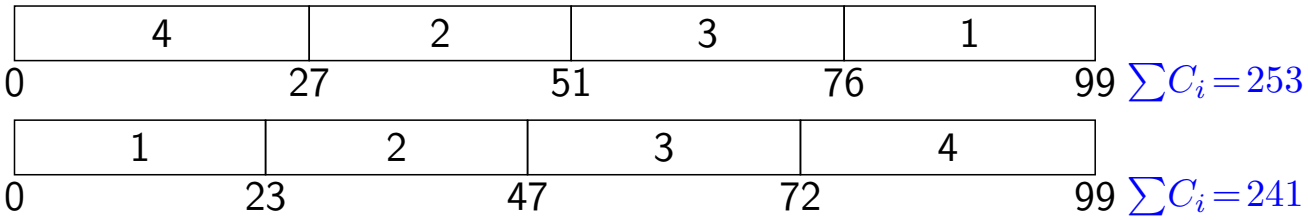
Exemple de problème d'optimisation déterministe

Ordonnancement de tâches sur une machine

Définitions

- J ensemble de tâches
- p_j, durée d'une tâche $j \in J$
- Ensemble de solutions X : toutes les séquences de tâches ($|X| = |J|!$)
- $C_i(x)$ date de fin d'une tâche dans la séquence x
- ORDO1MACH : trouver la séquence de tâche $x^* \in X$ qui minimise $\sum_{j \in J} C_j(x)$

Example : 4 jobs. $p_1 = 23$, $p_2 = 24$, $p_3 = 25$, $p_4 = 27$

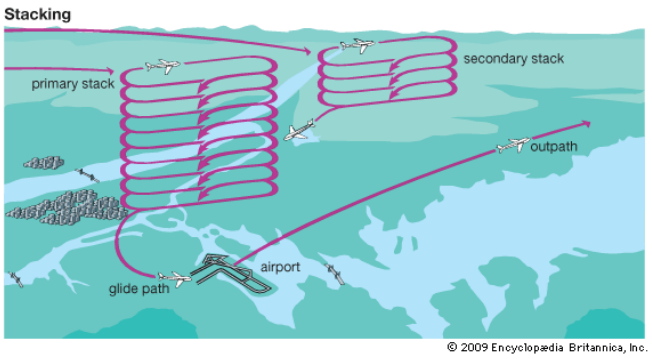


Exemple de problème d'optimisation déterministe

Ordonnancement des atterrissages dans un aéroport 1/2

Définitions

- P ensemble d'avions
- r_i date d'atterrissage au plus tôt
- d_i date d'atterrissage au plus tard
- t_i date d'atterrissage préférentielle
- s_{ij} temps de séparation minimal entre l'avion $i \in P$ et l'avion $j \in P$
- g_i (h_i) pénalité unitaire de retard (avance) par rapport à t_i



Problème : déterminer les dates d'atterrissage de chaque avion en respectant les fenêtres de temps, les distances de séparation et en minimisant la somme pondérée des avances et des retards

Exemple de problème d'optimisation déterministe

Ordonnancement des atterrissages dans un aéroport 2/2

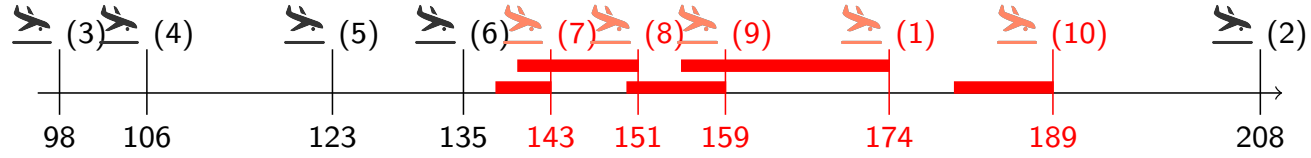
Exemple de problème : $|P| = 10$

(source http://people.brunel.ac.uk/~mastjjb/jeb/orlib/airlandinfo.html)

i	r_i	t_i	d_i	g_i	h_i
1	129	155	559	10	10
2	195	208	744	10	10
3	89	98	510	30	30
4	96	106	521	30	30
5	110	123	555	30	30
6	120	135	576	30	30
7	124	138	577	30	30
8	126	140	573	30	30
9	135	150	591	30	30
10	160	180	657	30	30

$$(s_{ij}) = \begin{pmatrix} - & 3 & 15 & 15 & 15 & 15 & 15 & 15 & 15 & 15 \\ 3 & - & 15 & 15 & 15 & 15 & 15 & 15 & 15 & 15 \\ 15 & 15 & - & 8 & 8 & 8 & 8 & 8 & 8 & 8 \\ 15 & 15 & 8 & - & 8 & 8 & 8 & 8 & 8 & 8 \\ 15 & 15 & 8 & 8 & - & 8 & 8 & 8 & 8 & 8 \\ 15 & 15 & 8 & 8 & 8 & - & 8 & 8 & 8 & 8 \\ 15 & 15 & 8 & 8 & 8 & 8 & - & 8 & 8 & 8 \\ 15 & 15 & 8 & 8 & 8 & 8 & 8 & - & 8 & 8 \\ 15 & 15 & 8 & 8 & 8 & 8 & 8 & 8 & - & 8 \\ 15 & 15 & 8 & 8 & 8 & 8 & 8 & 8 & 8 & - \end{pmatrix}$$

Exemple de solution:



Pénalité totale $= 30\times(189-180+159-150+151-140+143-138)+10\times(174-155) = 1210$

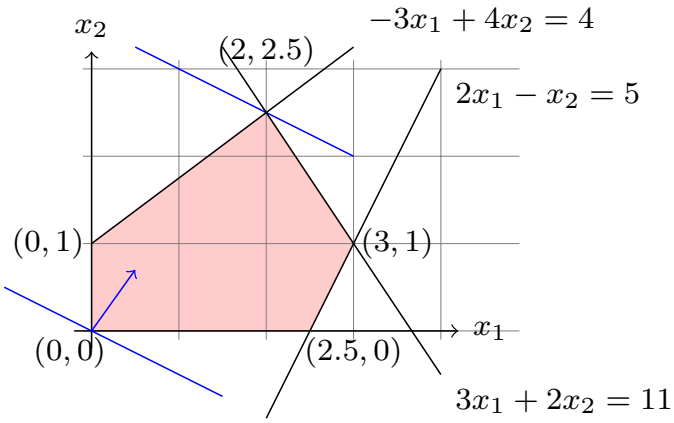
Outils pour l'optimisation déterministe

Programmation linéaire

Définitions

- Programmation linéaire = Problème d'optimisation continu avec contraintes et fonction objectif linéaires
- $f(\mathbf{x}) = c_1x_1 + c_2x_2 + \dots c_nx_n$
- $g_j(\mathbf{x}) = a_{1j}x_1 + a_{2j}x_2 + \dots + a_{nj}x_n - b_j,\ j = 1, \dots, m$
- PL : trouver $\mathbf{x}^* = \text{argmin}\{c^T\mathbf{x} | \mathbf{x} \in \mathbb{R}^n, A\mathbf{x} \geq \mathbf{b}\}$

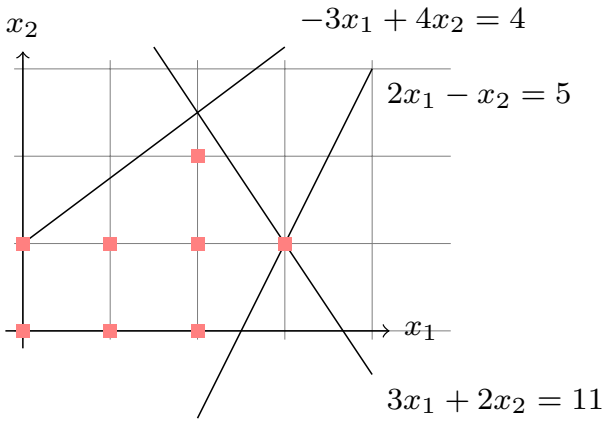
$$\begin{aligned} \max\ & x_1 + 2x_2 \\ -3x_1 + 4x_2 &\leq 4 \\ 3x_1 + 2x_2 &\leq 11 \\ 2x_1 - x_2 &\leq 5 \\ x_1, x_2 &\geq 0 \end{aligned}$$



Définitions

- Programmation linéaire avec variables entières
- PL : trouver $\mathbf{x}^* = \text{argmin}\{c^T\mathbf{x} | \mathbf{x} \in \mathbb{Z}^n, A\mathbf{x} \geq 0\}$

$$\begin{aligned} \max\ & x_1 + 2x_2 \\ -3x_1 + 4x_2 &\leq 4 \\ 3x_1 + 2x_2 &\leq 11 \\ 2x_1 - x_2 &\leq 5 \\ x_1, x_2 &\text{ entiers} \end{aligned}$$



Optimiser dans l'incertain

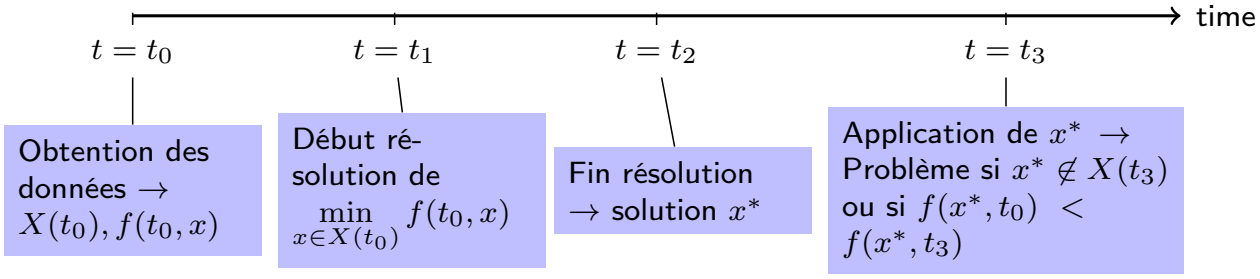
Les problèmes d'optimisation contiennent généralement des paramètres incertains.

L'objectif général de l'optimisation sous incertitudes est de trouver une solution pour un problème d'optimisation de telle sorte qu'elle soit robuste face aux incertitudes des paramètres du problème.

Remarques

- programmation linéaire sous incertitudes Dantzig, Charnes et al (années 50/60)
- optimisation robuste : Premiers travaux à la fin des années 90 : Ben-Tal, El Ghaoui, Nemirovski.
- Depuis 2000, une large littérature s'est développée.
- Les applications de la programmation robuste sont très importantes.

- Les données d'un problème d'optimisation sont en général connues de manière imprécise.
- Schématiquement la connaissance sur le problème à résoudre évolue dynamiquement au cours du temps : Ensemble des solutions $X(t)$ et objectif $f(x,t)$.
- La résolution du problème d'optimisation se fait donc à partir de données estimées au moment du calcul mais l'application des décisions est effectuée à un moment ultérieur et donc sur un problème différent.



- **Paramètres estimés**
  Certains paramètres sont estimés, par exemple par une analyse statistique.
  → Erreurs d'estimation
- **Paramètres mesurés**
  Certains paramètres sont difficilement mesurables avec certitude.
  → Erreurs de mesure

- **Paramètres révélés *a posteriori***
  Certains paramètres (demande, prix, etc.) ne sont révélés qu'après avoir fixé les variables de décision.
  $\rightarrow$ Erreurs de prévision
- **Mise en oeuvre**
  La solution calculée x ne peut pas être mise en oeuvre exactement comme prescrit, en raison de problèmes de précisions d'instrumentation ou d'erreurs humaines.
  $\rightarrow$ Erreurs de réalisation

Valeur moyenne : une bonne idée ?

Problème simple

$$\max\ x_2$$

s.c.

$$\left\{ \begin{array}{rcl} 10x_1 + ax_2 & \leq & 15 \end{array} \right.$$

Problème simple

$a = 5 \Rightarrow$ solution optimale $x^* = (0, 3)$

Valeur moyenne : une bonne idée ?

Problème simple

Considérons maintenant que $a = \bar{a} + \xi\tilde{a}$. Le paramètre incertain ξ suit une loi uniforme telle que $\xi \in [-1, 1]$.

Dans 50% des cas, la solution est infaisable !

Valeur moyenne : une bonne idée ?

A une échelle supérieure, si N contraintes sont actives dans un PL, la probabilité d'infaisabilité devient $1 - \left(\frac{1}{2}\right)^N$

Valeur moyenne : une bonne idée ?

Autre exemple

La contrainte suivante est active :

$$3.000x_1 - 2.999x_2 \leq 1$$

- La solution optimale est $x_1 = x_2 = 1000$.
- Le nombre 3.000 est incertain.

Quelles sont les conséquences d'une variation de +1% de ce nombre ?

Valeur moyenne : une bonne idée ?

Tests sur NETLIB

NETLIB est une collection d'une centaine de problèmes linéaires de tailles petites ou moyennes (du point de vue des standards actuels !) correspondant pour la plupart à des problèmes réels. La collection NETLIB a longtemps servi de banc d'essai pour comparer divers algorithmes (Simplex et Points Intérieurs).

Valeur moyenne : une bonne idée ?

Tests sur NETLIB

Exemple de contrainte dans le problème PILOT4 (contrainte 372)

$$
\begin{aligned}
a^T x \equiv\ & -15.79081x_{826} - 8.598819x_{827} - 1.88789x_{828} - 1.362417x_{829} - 1.526049x_{830} \\
& -0.031883x_{849} - 28.725555x_{850} - 10.792065x_{851} - 0.19004x_{852} - 2.757176x_{853} \\
& -12.290832x_{854} + 717.562256x_{855} - 0.057865x_{856} - 3.785417x_{857} - 78.30661x_{858} \\
& -122.163055x_{859} - 6.46609x_{860} - 0.48371x_{861} - 0.615264x_{862} - 1.353783x_{863} \\
& -84.644257x_{864} - 122.459045x_{865} - 43.15593x_{866} - 1.712592x_{870} - 0.401597x_{871} \\
& +x_{880} - 0.946049x_{898} - 0.946049x_{916} \\
\geq\ & b \equiv 23.387405
\end{aligned}
\qquad (C)
$$

Valeur moyenne : une bonne idée ?

$$
\begin{aligned}
a^T x \equiv\ & -15.79081x_{826} - 8.598819x_{827} - 1.88789x_{828} - 1.362417x_{829} - 1.526049x_{830} \\
& -0.031883x_{849} - 28.725555x_{850} - 10.792065x_{851} - 0.19004x_{852} - 2.757176x_{853} \\
& -12.290832x_{854} + 717.562256x_{855} - 0.057865x_{856} - 3.785417x_{857} - 78.30661x_{858} \\
& -122.163055x_{859} - 6.46609x_{860} - 0.48371x_{861} - 0.615264x_{862} - 1.353783x_{863} \\
& -84.644257x_{864} - 122.459045x_{865} - 43.15593x_{866} - 1.712592x_{870} - 0.401597x_{871} \\
& +x_{880} - 0.946049x_{898} - 0.946049x_{916} \\
\geq\ & b \equiv 23.387405
\end{aligned}
\qquad (C)
$$

Certains coefficients semblent "douteux". Il est peu réaliste d'avoir une précision aussi importante (9 chiffres significatifs).

Supposons que les coefficients douteux sont connus avec une précision de 0.1%.

Valeur moyenne : une bonne idée ?

Dans le pire cas la contrainte est violée de 450% : $Ax = 4.5b$

Des tests ont été réalisés avec 1000 tirages aléatoires avec $a_i^{true} = \bar{a}_i + \epsilon_i |a_i|$ et $\epsilon_i \sim [-0.001, 0.001]$.

- La contrainte est violée dans 50% des cas.
- Dans 18% des cas, elle est violée de 150%
- En moyenne elle est violée de 125%

Valeur moyenne : une bonne idée ?

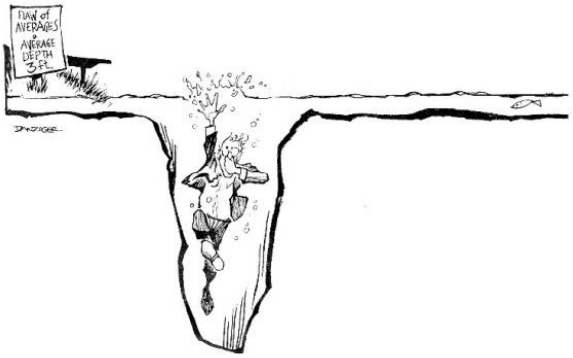
Résultats des tests sur NETLIB

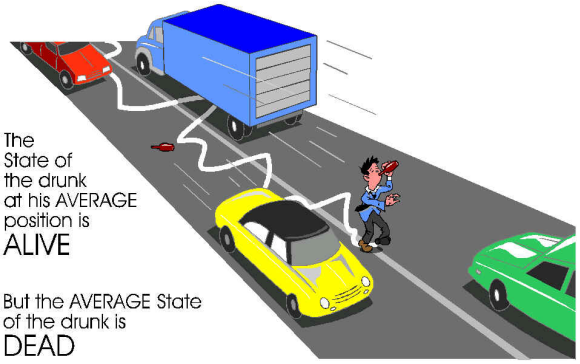
Sur les 90 problèmes de la collection

- 27 problèmes ont une solution nominale non fiable au niveau de perturbation 1%
- parmi ces 27 problèmes, 19 sont «mauvais» au niveau d'incertitude 0.01%
- pour 13 de ces 19 problèmes, une perturbation de $\epsilon = 0.01\%$ peut entraîner une violation de plus de 50% pour certaines contraintes.

Incertitudes dans le modèle

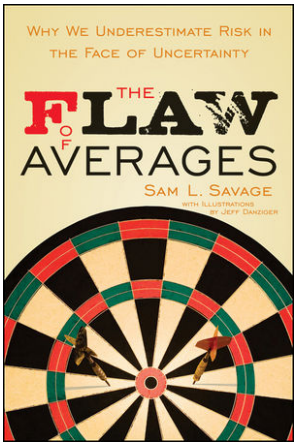
Que se passe t-il si l'on considère des valeurs moyennes pour les paramètres incertains ?





Incertitudes dans le modèle

"Plans based on average assumptions are wrong, on average !"



Incertitudes dans le modèle

Conclusion sur l'approche nominale

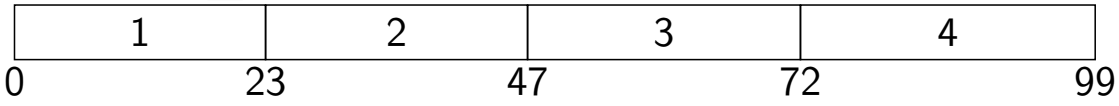
En programmation mathématique, on ne peut pas se contenter des valeurs nominales car des petites variations sur les données peuvent rendre les solutions nominales irréalisables.

Incertitudes sur les données

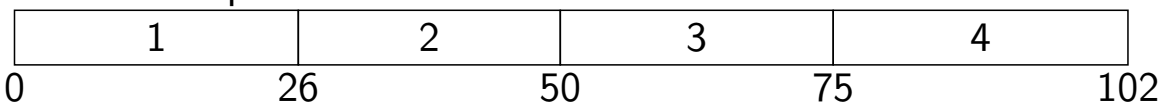
Impact des incertitudes

Une petite variation des données peut avoir un grand impact sur l'objectif !

Ordonnancement "optimal" avec $p_1 = 23$, $p_2 = 24$, $p_3 = 25$, $p_4 = 27$ de coût estimé $\sum C_i = 241$.

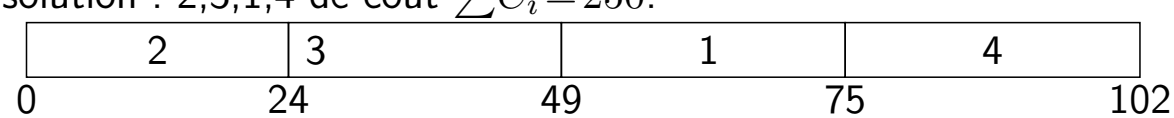


L'exécution de l'ordonnancement révèle une augmentation de p_1 de 3 unités de temps.



Le coût réel de la solution dépasse le coût estimé de 12 unités de temps $(\sum C_i = 253)$!

Des regrets ? Oui car pour les données observées, il existait une meilleure solution : 2,3,1,4 de coût $\sum C_i = 250$.



Bonnes pratiques d'optimisation sous incertitudes

Optimiser au bon moment

Lancer le calcul d'optimisation le plus tard possible pour avoir le problème le plus proche possible de la réalité ? Oui, mais :

- Pour des raisons de logistique, la solution doit parfois être fournie suffisamment en avance par rapport à son application.
- Cette méthode permet d'éliminer certaines (mais pas toutes les) erreurs de prévisions mais pas les erreurs de mesures ni d'implémentation.

Bonnes pratiques d'optimisation sous incertitudes

Réoptimiser

Modifier la solution une fois que les données réelles sont révélées, en relançant la résolution ? Oui, mais :

- Certaines données ne sont révélées qu'après l'application de la solution (une tâche dont on ne connait la durée réelle qu'une fois terminée, un revenu qu'on ne connait qu'une fois le produit fabriqué, une erreur de mesure...).
- On peut parfois appliquer une partie de la solution, et relancer le calcul d'optimisation pour le reste des décisions, selon la procédure **d'horizon glissant**.

Bonnes pratiques d'optimisation sous incertitudes

Réoptimiser au bon moment : l'optimisation par horizons glissants

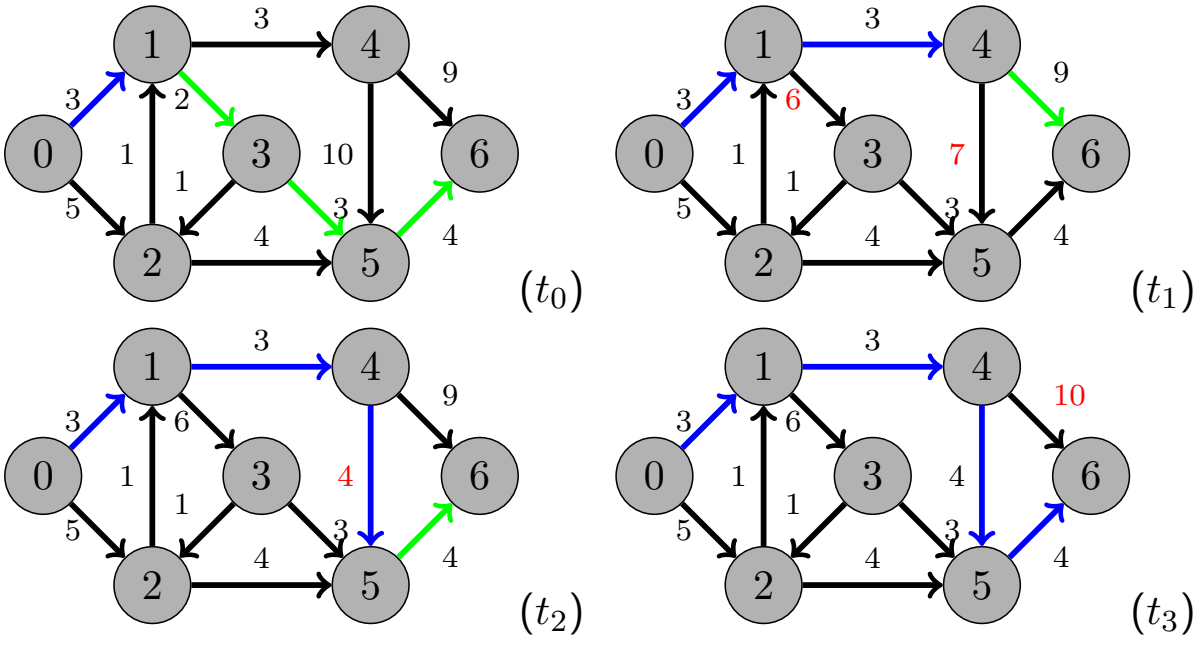
- En pratique, seulement une partie des décisions représentées par la solution x^* sont à appliquer impérativement avant une certaine date.
- Sur la base des deux bonnes pratiques précédentes, un schéma général multi-étapes peut être énoncé.
  - On suppose qu'une solution x^* est décomposable en décisions élémentaires.
  - La procédure d'optimisation par horizon glissant applique récursivement le principe suivant :

Algorithm 1 Optimisation par Horizon Glissant

1: Déterminer une date t avant laquelle un sous ensemble de décisions d doit être déterminé
2: Résoudre le problème d'optimisation $P(t')$ avec $t' \leq t$ pour que la solution x^* soit disponible avant t
3: Appliquer les décisions d à partir de x^* et réitérer la procédure

Exemple d'optimisation par horizon glissant

On fixe le premier arc (en bleu) du plus court chemin (en vert) après chaque calcul.



Modélisation des incertitudes

Quelle connaissance a-t-on à l'avance des paramètres ?

- Parfaite → Optimisation **déterministe**.
- Aucune → Optimisation **en ligne** (*online*).
- Probabiliste → Optimisation **stochastique**.
- Par scénarios → Optimisation **robuste**.

En pratique, les 4 cas de figure peuvent se produire en même temps sur un même problème !

Optimisation en ligne

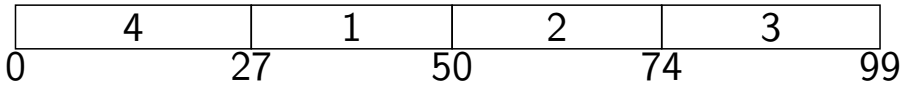
- Le problème est inconnu au départ et est révélé au cours du temps.
- A partir du moment où les paramètres du problème sont révélés, la décision doit être prise de manière urgente et sans connaissance du futur.

Exemple

Ordonnancement en ligne. L'ensemble des tâches J est inconnu au départ. Chaque tâche arrive à une date r_i. Le choix est limité aux tâches déjà arrivées. Algorithme en-ligne : prendre la tâche de durée la plus petite.

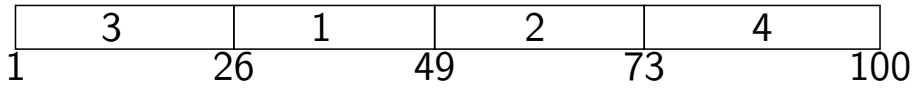
$p_1 = 23$, $r_1 = 2$, $p_2 = 24$, $r_2 = 2$, $p_3 = 25$, $r_3 = 1$, $p_4 = 27$, $r_4 = 0$

Ordonnancement en-ligne :



$\sum C_i = 250$

Ordonnancement optimal :



$\sum C_i = 248$

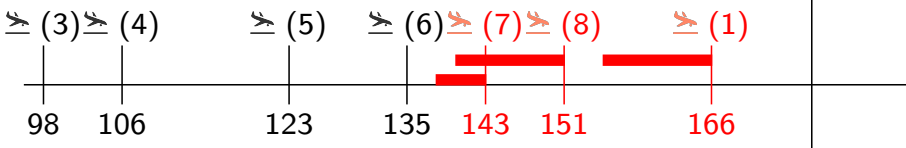
Optimisation en ligne

Exemple de l'atterrissage d'avions

- En réalité, la planification de l'atterrissage d'un avion n'est considérée qu'à partir du moment où l'avion apparait dans la zone de contrôle.
- Soit a_i la date d'apparition d'un avion.
- A chaque date arrivée $T = a_i$ d'un avion, on ne peut planifier que les avions $j \in P$ qui sont déjà apparus ($a_j \leq T$) et pour lesquels l'atterrisage planifié S_j n'est pas trop proche ($S_j \leq T - f$) où f est la période de "gel" de la planification.

i	a_i	r_i	t_i	d_i	g_i	h_i
1	54	129	155	559	10	10
2	120	195	208	744	10	10
3	14	89	98	510	30	30
4	21	96	106	521	30	30
5	35	110	123	555	30	30
6	45	120	135	576	30	30
7	49	124	138	577	30	30
8	51	126	140	573	30	30
9	60	135	150	591	30	30
10	85	160	180	657	30	30

solution à $t = 54$:



.. doit être remise à cause à $t = 60$ (arrivée vol 9 avec une date préférentielle d'atterrissage inférieure à t_1

Optimisation en ligne

Facteur de compétitivité

- **Instance** d'un problème : données d'un problème d'optimisation et de toutes les valeurs numériques des paramètres.
- **Facteur de compétitivité** sur une instance I (problème de minimisation) : valeur $f(I, A)$ obtenue par l'algorithme en ligne A sur l'instance I divisée par la valeur optimale $f^*(I)$ pour l'instance I.

$$\rho(A, I) = f(I, A)/f^*(I)$$

Exemple pour l'instance précédente $\rho(A, I) \simeq 1,02$.

- **Facteur de compétitivité dans le pire des cas** : pire facteur de compétitivité sur l'ensemble des instances $\mathcal{I}$

$$\rho(A) = \max_{I \in \mathcal{I}} f(I, A)/f^*(I)$$

Objectif : obtenir un algorithme en ligne du meilleur facteur de compétitivité dans le pire des cas possible.

En étudiant souhaite aller skier dans les Pyrénées pour la première fois. S'équiper complètement lui coûterait B (trentaines d'Euros) alors que louer l'équipement lui coûterait 1 (trentaine d'Euros) la journée. On suppose que l'étudiant n'a aucune idée du nombre de jours (D) où il ira skier par la suite. Il s'agit donc d'un problème d'optimisation sous incertitude. A chaque fois que l'étudiant va skier, s'il n'a pas déjà acheté il doit décider s'il achète ou s'il loue pour minimiser un coût qui ne sera connu qu'une fois le nombre de jours de ski révélé.

Donner le facteur de compétitivité dans le pire des cas d'une politique consistant pour l'étudiant à acheter l'équipement la k-ème fois qu'il va skier. En déduire la meilleure politique en termes de compétitivité dans le pire des cas.

Cas déterministe (trivial)

On connait parfaitement le nombre de jours de ski D

L'algorithme de détermination de la solution optimale est

Si $D < B$ alors
  Louer jusqu'à D et ne pas acheter. Coût $= D$
Sinon Si $D \geq B$ alors
  Acheter dès le premier jour. Coût=B
Fin Si

Et le coût optimal est $\min(D, B)$

Optimisation en ligne

Exercice : acheter ou louer ?

Le nombre de jour de ski D est totalement inconnu

Un algorithme online consiste à décider uniquement en fonction des données connues (le prix de la location et le prix de vente) et du nombre de jour de ski déjà effectué.
Par exemple, on peut fixer un nombre maximum de jours de ski K avant d'acheter. L'algorithme en ligne $A(K)$ est alors un algorithme de prise de décision avant chaque jour de ski k:

Si $k < K$ alors
  Louer. Coût = Coût + 1
Sinon Si $k = K$ alors
  Acheter. Coût=Coût+B
Fin Si

Optimisation en ligne

Exercice : acheter ou louer ?

Le coût total obtenu par l'algorithme en ligne, en fonction du nombre de jours de ski effectivement réalisé D (l'instance est définie par la valeur D) est

$$f(D, A(K)) = \begin{cases} D & \text{si} \quad D < K \\ K - 1 + B & \text{si} \quad D \geq K \end{cases}$$

Etant donné cette politique le pire des cas est que K soit la dernière journée de ski ($D = K$). Le coût est alors toujours de $f(K, A(K)) = K - 1 + B$.

si $K \leq B$: l'optimum était de louer $f^*(K) = K$. Facteur de compétitivité $\frac{K-1+B}{K}$
si $K \geq B$: l'optimum était d'acheter ($f^*(K) = B$). Facteur de compétitivité $\frac{K-1+B}{B}$

En résumé

$$\rho(A(K)) = \rho(A(K), K) = \begin{cases} 1 + \frac{B-1}{K} & \text{si} \quad K \leq B \\ \frac{K+B-1}{B} & \text{si} \quad K \geq B \end{cases}$$

On cherche l'algorithme $A(K)$ de plus petit facteur de compétitivité. On a :

$$min\{1 + \frac{B-1}{K} | K = 1, ..., B\} = 1 + \frac{B-1}{B} = 2 - \frac{1}{B}$$

et

$$min\{\frac{K+B-1}{B} | k >= B\} = \frac{2B-1}{B} = 2 - \frac{1}{B}$$

L'algorithme de meilleur facteur de compétitivité consiste donc à acheter à la Bème journée:

$$\rho(A(B)) = 2 - \frac{1}{B} < 2$$

.

Optimisation stochastique

- Les paramètres du problèmes d'optimisation varient selon des lois de probabilité.

Examples :

- les coûts des arcs l_{ij} dans le problème du plus court chemin $\rightarrow \tilde{l}_{ij}$
- les durées des tâches p_i dans le problème d'ordonnancement $\rightarrow \tilde{p}_i$
- le vecteur de coût c, la matrice A et le second membre b en programmation linéaire $\rightarrow \tilde{c}$, $\tilde{A}$, $\tilde{b}$.

On minimize généralement l'espérance de la fonction objectif.

$$x^* = \underset{x \in \tilde{X}}{\text{argmin}} \, E[\tilde{f}(x)]$$

avec $\tilde{X}$ ensemble de solutions décrit par des lois de probabilités et $\tilde{f}$ la fonction objectif décrite par des lois de probabilité.

Optimisation stochastique

Exemple en ordonnancement

On considère que le problème d'ordonnancement est tel que les durées des tâches varient de manière mutuellement indépendante selon des lois uniformes :

$$\tilde{p}_1 \sim \mathcal{U}[23, 24] \quad \tilde{p}_2 \sim \mathcal{U}[21, 27] \quad \tilde{p}_3 \sim \mathcal{U}[20, 29] \quad \tilde{p}_4 \sim \mathcal{U}[5, 45]$$

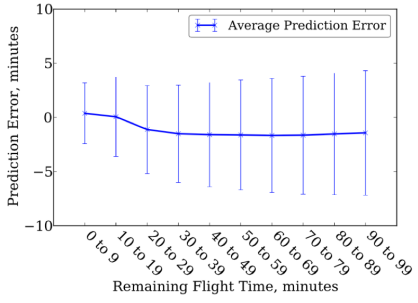
Problème d'ordonnancement stochastique : trouver l'ordonnancement qui minimise l'espérance de la somme des dates de fin des tâches $E[\sum_{j \in J} C_i]$.

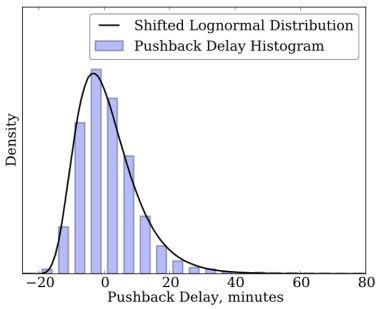
La solution optimale consiste à ordonnancer les tâches selon l'ordre de la plus petite durée espérée, soit ici $1 - 2 - 3 - 4$ car $E[p_1] = 23,5$, $E[p_2] = 24$, $E[p_3] = 24,5$, $E[p_4] = 25$ avec $E[\sum_{j \in J} C_i] = 240$.

Optimisation stochastique

Le problème d'ordonnancement d'atterrissages stochastique

- la date d'arrivée d'un avion est rarement égale à la date prévue





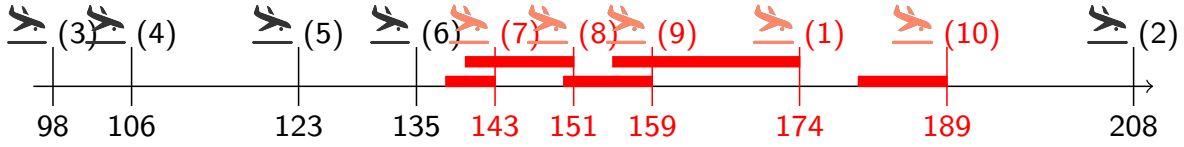
source : https://smartech.gatech.edu/bitstream/handle/1853/44842/solveling_gustaf_h_201208_phd.pdf

- Distribution de probabilité. Exemple: modélisation de la date d'arrivée $\tilde{r}_i$ comme une variable aléatoire discrète selon des scénarios S donnant pour chaque avion une date r_{is} avec la probabilité π_{is} pour chaque scénario $s \in S$.
- Nécessité de définir une politique d'ordonnancement étant donné une séquence (ex: atterrissage au plus tôt)
- Trouver la séquence qui minimise l'espérance de la pénalité

Optimisation stochastique

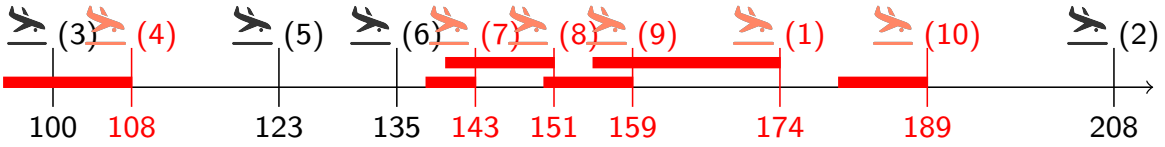
Le problème d'ordonnancement d'atterrissages stochastique

- Séquence $3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 7 \rightarrow 8 \rightarrow 9 \rightarrow 1 \rightarrow 10 \rightarrow 2$
- Scenario 1 $(r_{i1}) = (t_{i1}) = (155, 208, 98, 106, 123, 135, 138, 140, 150, 180)$, $\pi_s = 0,4$



Pénalité 1210

- Scenario 2 $(r_{i2}) = (t_{i2}) = (155, 208, 100, 95, 123, 135, 138, 140, 150, 180)$, $\pi_s = 0,6$



Pénalité 1600

- Pénalité moyenne $1210 \times 0,4 + 1600 \times 0,6 = 1444$

Optimisation stochastique

Le problème de location de ski (version stochastique)

Le nombre de jours de ski est maintenant une variable aléatoire discrete $\tilde{D}$ telle que $P[\tilde{D} = q] = \pi_q$, $\forall q \in \mathbb{N}$ avec $\sum_{q \in \mathbb{N}} \pi_q = 1$.

Ecrivons l'espérance du coût de la décision d'acheter au jour K.

Rappel: si le nombre de jours de ski réalisé D est strictement plus petit que K, on paye D, sinon on paye $K - 1 + B$. En terme d'espérance, on obtient:

$$E[\tilde{f}(K)] = \sum_{q=1}^{K-1} q\pi_q + \sum_{q=K}^{+\infty} (K - 1 + B)\pi_q$$

Considérons que $\tilde{D} \sim \mathcal{U}[a, b]$ avec $B \in [a, b]$. On a $\pi_q = \frac{1}{b-a+1}$ pour $q \in [a, b]$ et $\pi_q = 0$ pour $q \in [1, a-1] \cup [b+1, +\infty[$.

On obtient trois cas possibles :

Si $K \leq a-1$, $E[\tilde{f}(K)] = K - 1 + B$

Si $K \geq b+1$, $E[\tilde{f}(K)] = E[\tilde{D}] = \frac{a+b}{2}$

Si $a \leq K \leq b$,

$$E[\tilde{f}(K)] = \sum_{q=a}^{K-1} \frac{q}{b-a+1} + \sum_{q=K}^{b} \frac{K-1+B}{b-a+1}$$

Cas $a \leq K \leq b$

On remarque que

$$\sum_{q=a}^{K-1} \frac{q}{b-a+1} + \sum_{q=K}^{b} \frac{K-1+B}{b-a+1} = \begin{cases} \sum_{q=a}^{b} \frac{K-1+B}{b-a+1} = (K-1+B) & \text{si K=a} \\ \frac{\sum_{q=1}^{K-1} q - \sum_{q=1}^{a-1} q + (K-1+B)(b-K+1)}{b-a+1} & \text{sinon} \end{cases}$$

On rappelle que $\sum_{i=1}^{j} i = \frac{j(j+1)}{2}$. D'où pour $K = a+1, \ldots, b$ on obtient :

$$\begin{aligned} E[\tilde{f}(K)] &= \frac{K(K-1) - a(a-1) - 2(b-K+1)(K-1+B)}{2(b-a+1)} \\ &= \frac{-K^2 + (2b-2B+3)K + 2bB - 2b - a^2 + a - 2 + 2B}{2(b-a+1)} \end{aligned}$$

Il faut donc minimiser un polynôme de degré 2 en K (entier) pour trouver l'espérance minimum.

Application numérique : $a = 5$, $b = 10$, $B = 8$

On obtient $K - 1 + B = 8$ et $(b + a)/2 = 7.5$

$E[\tilde{f}(K)] = \frac{-K^2+7K+134}{12}$ pour $K \in [6, 10]$

D'où :

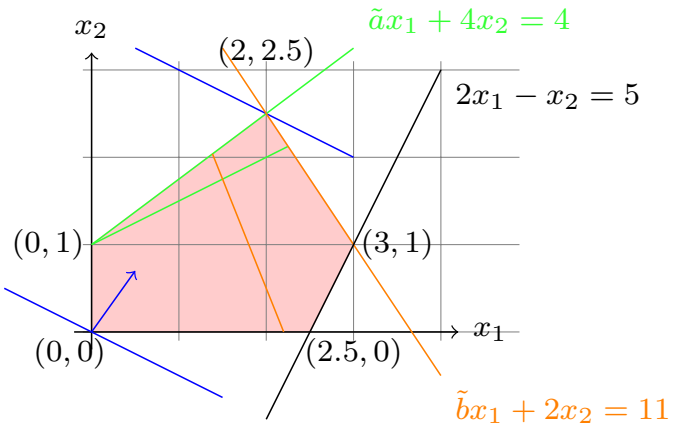
Pour $K = 1$, $E[\tilde{f}(K)] = 8$
Pour $K = 2$, $E[\tilde{f}(K)] = 9$
Pour $K = 3$, $E[\tilde{f}(K)] = 10$
Pour $K = 4$, $E[\tilde{f}(K)] = 11$
Pour $K = 5$, $E[\tilde{f}(K)] = 12$
Pour $K = 6$, $E[\tilde{f}(K)] = 11.6667$
Pour $K = 7$, $E[\tilde{f}(K)] = 11.1667$
Pour $K = 8$, $E[\tilde{f}(K)] = 10.5$
Pour $K = 9$, $E[\tilde{f}(K)] = 9.66667$
Pour $K = 10$, $E[\tilde{f}(K)] = 8.66667$
Pour $K >= 11$, $E[\tilde{f}(K)] = 7.5$

La solution optimale est de toujours louer.

- Dans l'exemple en ordonnancement, toute séquence donne une solution réalisable, quel que soit la réalisation des paramètres aléatoires. Dans beaucoup de problèmes, une solution définie avant la connaissance des valeurs réalisées peut être irréalisable. Exemple en programmation linéaire :

$$\begin{aligned} \max\ & x_1 + 2x_2 \\ & \tilde{a}x_1 + 4x_2 \leq 4 \\ & \tilde{b}x_1 + 2x_2 \leq 11 \\ & 2x_1 - x_2 \leq 5 \\ & x_1, x_2 \geq 0 \end{aligned}$$

avec $\tilde{a} \sim \mathcal{U}[-3, -2]$ et $\tilde{b} \sim \mathcal{U}[3, 5]$



On considère de contraindre la *probabilité* de satisfaction des contraintes au niveau souhaité.

Reformulation du programme linéaire

$$\text{(PL)} \qquad \min z$$
$$c_1 x_1 + c_2 x_2 + \ldots + c_n x_n \leq z$$
$$a_{1j} x_1 + a_{2j} x_2 + \ldots + a_{nj} x_n \geq b_j \qquad j = 1, \ldots, m$$
$$x_i \geq 0 \qquad i = 1, \ldots, n$$

Programme linéaire contraint en probabilité

$$\text{(PLCP)} \qquad \min z$$
$$\mathbb{P}\{\tilde{c}_1 x_1 + \tilde{c}_2 x_2 + \ldots + \tilde{c}_n x_n \leq z\} \geq \alpha$$
$$\mathbb{P}\{\tilde{a}_{1j} x_1 + \tilde{a}_{2j} x_2 + \ldots + \tilde{a}_{nj} x_n \geq \tilde{b}_j\} \geq \beta_j \qquad j = 1, \ldots, m$$
$$x_i \geq 0 \qquad i = 1, \ldots, n$$

Exemple de la location de ski. Quel cout maximum peut-on garantir avec une probabilité β ?

$$\min V$$
$$P[\tilde{f}(K) \leq V] \geq \beta$$

On prendra $\tilde{D} \sim \mathcal{U}[a, b]$ avec $B \in [a, b]$ et l'application numérique $B = 8$, $a = 5$ et $b = 10$.

Retour sur l'exemple

$$\begin{aligned} \max\, & x_1 + 2x_2 \\ & \mathbb{P}\{\tilde{a}x_1 + 4x_2 \leq 4\} \geq \alpha \\ & \mathbb{P}\{\tilde{b}x_1 + 2x_2 \leq 11\} \geq \beta \\ & 2x_1 - x_2 \leq 5 \\ & x_1, x_2 \geq 0 \end{aligned}$$

Remarque. On peut diminuer le risque en considérant des probabilités jointes :

$$\begin{aligned} \max\, & x_1 + 2x_2 \\ & \mathbb{P}\{\tilde{a}x_1 + 4x_2 \leq 4, \tilde{b}x_1 + 2x_2 \leq 11\} \geq \alpha \\ & 2x_1 - x_2 \leq 5 \\ & x_1, x_2 \geq 0 \end{aligned}$$

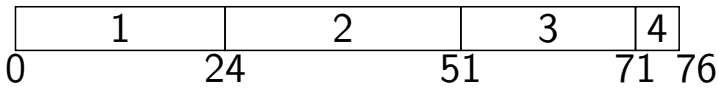
- Il n'est pas toujours facile ni même possible d'obtenir les lois de probabilité des paramètres incertains, encore moins des lois de probabilités indépendantes.
- En pratique, les décideurs peuvent vouloir être très prudent et demander que la décision soit *robuste*, i.e. qu'elle soit la meilleure (ou moins mauvaise) possible si le pire scénario se produit, ce que n'assure pas une optimisation de l'espérance de la fonction objectif.

Exemple : Retour sur le problème d'ordonnancement

$$\tilde{p}_1 \sim \mathcal{U}[23, 24] \quad \tilde{p}_2 \sim \mathcal{U}[21, 27] \quad \tilde{p}_3 \sim \mathcal{U}[20, 29] \quad \tilde{p}_4 \sim \mathcal{U}[5, 45]$$

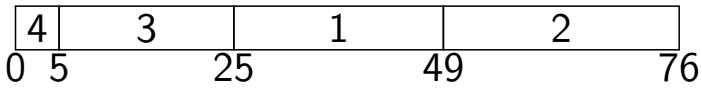
La solution qui minimise l'espérance du critère de somme des dates de fin consiste à ordonnancer les tâches selon l'ordre de la plus petite durée espérée, soit ici $1 - 2 - 3 - 4$ car $E[p_1] = 23,5$, $E[p_2] = 24$, $E[p_3] = 24,5$, $E[p_4] = 25$ avec $E[\sum_{j \in J} C_i] = 240$.

Exemple : Retour sur le problème d'ordonnancement (suite)
Considérons la réalisation $p_1 = 24$, $p_2 = 27$, $p_3 = 20$, $p_4 = 5$. En appliquant la solution optimale du problème d'ordonnancement stochastique, on obtient la solution ::



$\sum C_i = 222$

Considérons maintenant la solution optimale pour cette réalisation (ordre des plus petites durées) :



$\sum C_i = 155$

La solution "optimale' ' stochastique excède de 67 unités de temps ou de $43,2\%$ la solution optimale pour la réalisation considérée !

- Principe : obtenir des garanties **dans le pire des cas** pour une modélisation raisonnable des incertitudes (sous la forme de scénarios)

Définitions préalables

- S : ensemble (continu ou discret) de scénarios.
- D^s : réalisation des paramètres du problème pour un scénario donné $s \in S$.
- X^s : ensemble des solutions réalisables pour un scénario donné $s \in S$
- $f : X \times S$ fonction objectif : $f(x, s)$ valeur de l'objectif pour le scénario $s \in S$ et la solution $x \in X^s$

Problème d'optimisation robuste (critère minimax)

Trouver $x^* = \text{argmin}_{x \in \cap_{s \in S} X^s} \max_{s \in S} f(x, s)$

On remplace la modélisation des incertitudes sous forme de loi uniforme par les intervalles correspondant à chaque loi :

$$p_1 \in [23, 24] \quad p_2 \in [21, 27] \quad p_3 \in [20, 29] \quad p_4 \in [5, 45]$$

Pour toute séquence de tâche le pire scénario pour la minimisation des dates de fin est clairement celui qui donne la durée maximale à chaque tâche.
Pour ce scénario, la séquence optimale est $1 - 2 - 3 - 4$ selon la règle des plus petites durées. C'est donc la séquence optimale pour le problème minimax. On retrouve dans ce cas la même solution que pour le cas stochastique.

Principe : Lorsque la solution appliquée est évaluée a posteriori (après la réalisation des données) un décideur peut être intéressé par la minimisation de l'écart dans le pire des cas entre la valeur de la solution qu'il applique et la valeur qu'il aurait pu obtenir s'il avait connu le scénario au départ. C'est le concept de regret $\rho(x, s)$ d'une solution x pour le scénario s

$$\rho(x, s) = f(x, s) - \min_{y \in X^s} f(y, s)$$

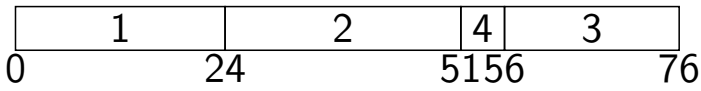
Problème d'optimisation robuste (critère minimax regret absolu)

Trouver $x^* = \text{argmin}_{x \in \cap_{s \in S} X^s} \max_{s \in S} \rho(x, s)$

Problème d'optimisation robuste (critère minimax regret relatif)

Trouver $x^* = \text{argmin}_{x \in \cap_{s \in S} X^s} \max_{s \in S} \left(\frac{\rho(x,s)}{\min_{y \in X^s} f(y,s)} \right)$

Revenons au scénario $p_1 = 24$, $p_2 = 27$, $p_3 = 20$, $p_4 = 5$, on obtient en appliquant la séquence $1 - 2 - 4 - 3$:



$\sum C_i = 207$

Le regret absolu de cette solution pour ce scénario est de 207-155=52, à comparer avec le regret absolu de la solution optimale stochastique (ou minimax) égal à 222-155=67.

Le regret relatif de cette solution est de $52/155 \simeq 33,54\%$ à comparer avec le regret relatif maximal de la solution optimale stochastique (ou minimax) égal à $67/155 \simeq 42,22\%$.

Les approches d'optimisation robuste se distinguent par les différentes modélisation des incertitudes.

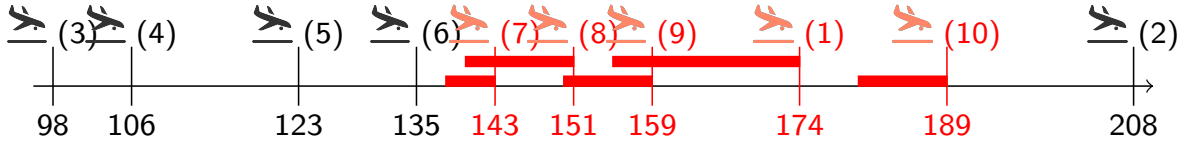
- Les scénarios sont donnés explicitement, chaque scénario donnant les valeurs des paramètres incertains pour ce scénario.
- Chaque paramètre incertain et donné sous la forme d'un intervalle de valeurs possibles $[a, b]$ et toutes les combinaisons des valeurs des paramètres sont également possibles.
- Chaque paramètre incertain et donné sous la forme d'une valeur nominale v et d'un écart possible ϵ, mais on suppose qu'au plus k paramètres peuvent s'écarter de leur valeur nominale.
- Autres modélisations (incertitudes ellipsoidales, polyhédrales, ...)

Optimisation robuste

Le problème d'ordonnancement d'atterrissages robuste

- Séquence (x) $3 \to 4 \to 5 \to 6 \to 7 \to 8 \to 9 \to 1 \to 10 \to 2$
- Scenario $s = 1$

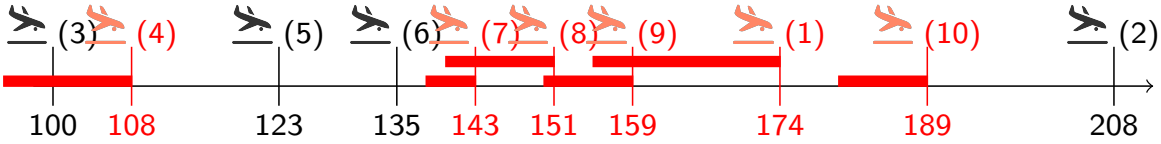
  $(r_{i1}) = (t_{i1}) = (155, 208, 98, 106, 123, 135, 138, 140, 150, 180)$



  Pénalité 1210

- Scenario $s = 2$

  $(r_{i2}) = (t_{i2}) = (155, 208, 100, 95, 123, 135, 138, 140, 150, 180)$



  Pénalité 1600

- coût $\max_{s \in S} f(x, s) = \max(1210, 1600) = 1600$

Optimisation Robuste

Exercice : Le problème de location de ski robuste

On sait qu'on va skier soit B-3 jour soit B+2 jour en supposant que B>5. Exemple B = 8 et on va skier soit une semaine (5 jours) soit deux semaines (10 jours).

Décisions possibles : acheter dès le début ou louer jusqu'au bout ?

Objectif 1 : minimiser la valeur du cout dans le pire des cas
si on loue on paye au pire B+2
si on achète dès le début on paye au pire B : décision optimale pour minimiser le pire des cas selon les scénarios.

Objectif 2 : minimiser le regret absolu dans le pire des cas
acheter dès le début regret max = B - B + 3 = 3
louer jusqu'au bout : regret max = B+2 - B = 2

Objectif 3 : minimiser le regret relatif dans le pire des cas
acheter dès le début regret max = (B - B + 3) / (B-3) ex pour B = 8 : 3/5 = 0,6
louer jusqu'au bout : regret max = (B+2 - B) / B ex pour B = 8 : 2/8 = 0,25

Optimisation Robuste

Exercice : Le problème du marchand de journaux robuste

Un marchand de journaux souhaite déterminer le nombre d'exemplaires Q d'un journal qu'il doit commander. Il ne connait pas la demande précise mais sait qu'elle est comprise entre deux valeurs $[\underline{d}, \overline{d}]$. Chaque exemplaire commandé lui coûte v €. Chaque exemplaire vendu lui rapporte p €. Chaque exemplaire retourné comme invendu lui rapporte g €. Il paye un coût B de perte potentielle de clientèle pour chaque demande non satisfaite.

Calculer l'expression du profit en fonction de la demande d et de la quantité commandée Q.
Résoudre les problèmes d'optimisation robuste avec les critères maximin, minimax regret absolu et minimax regret relatif.

Algorithme d'optimisation en ligne

- L'algorithme A doit servir séquentiellement un ensemble de *requêtes* $\sigma(1), \sigma(2), \ldots, \sigma(m)$
- "Servir" la requête i revient à prendre de manière définitive une décision sous la forme d'une *réponse* $\delta_i \in D_i$, ou D_i est l'ensemble des réponses possibles
- La séquence de décisions $(\delta_1, \delta_2, \ldots, \delta_m)$ donne la solution d'une instance [1] $I = (\sigma_i)_{i=1,\ldots,m}$ d'un problème d'optimisation progressivement révélée par les "reqûetes".
- L'ensemble des décisions prises a ainsi un coût $f^m(\delta_1, \delta_2, \ldots, \delta_m)$ qu'il s'agit de minimiser, $f^i(\delta_1, \delta_2, \ldots, \delta_i)$ donnant le coût intermédiaire des décisions $\delta_1, \ldots, \delta_i$.
- Mais chaque requête $\sigma(i)$ doit être servie (et donc la décision δ_i doit être prise) en ligne sans connaissance du futur, i.e. les requêtes $\sigma(i+1), \ldots, \sigma(m)$.

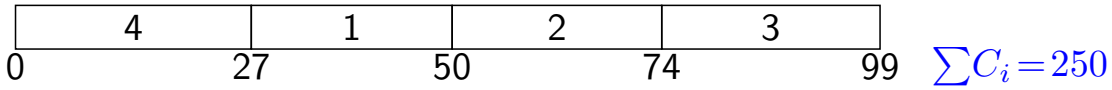
1. Instance : donnée d'un problème d'optimisation et de toutes les valeurs numériques des paramètres

Algorithme d'optimisation en ligne

Différents modèles d'ordonnancement en ligne

- Modèle d'arrivée des tâches
  - Chaque tâche arrive à une date précise, inconnue au départ (modèle de temps : décision à prendre à chaque date d'arrivée, en temps réel)
  - Chaque tâche arrive dans un ordre inconnu au départ (modèle de séquence ou de liste : décision à prendre à chaque apparition dans la liste)
- Modèle de connaissance des tâches une fois arrivées
  - Clairvoyance : les paramètres d'une tâche sont connus à son arrivée
  - Non clairvoyance : les paramètres de la tâche restent inconnus jusqu'à la fin de la tâche

Exemple de modèle de temps & clairvoyant: 4 tâches de durées $p_1 = 23$, $p_2 = 24$, $p_3 = 25$, $p_4 = 27$ et de dates d'arrivées $r_1 = 2$, $r_2 = 2$, $r_3 = 1$, $r_4 = 0$.
Algorithme en ligne SPT: t=0 tâche 4 arrivée, t=27 tâches 1,2,3 arrivées

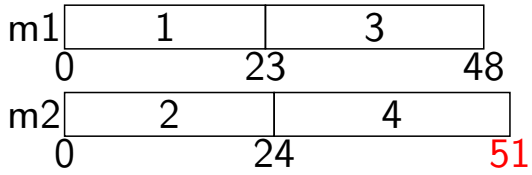


$\sum C_i = 250$

Algorithme d'optimisation en ligne

Ordonnancement sur machines parallèles

Exemple de modèle de liste & clairvoyant: 4 tâches de durées $p_1 = 23$, $p_2 = 24$, $p_3 = 25$, $p_4 = 27$, 2 machines parallèles.
Les tâches arrivent dans l'ordre 1,2,3,4
Objectif minimiser la durée totale d'ordonnancement $C_{\max}$
Algorithme en ligne : Ordonnancement la tâche sur la machine qui se libère le plus tôt



Analyse de compétitivité

- Soit $f^*(I) = \min\{f^m(\delta) | \delta \in D_1 \times D_2 \times \ldots \times D_m\}$ la valeur optimale compte tenu des décisions possibles à chaque étape.
- Soit $f(I, A) = f^m(\delta^A)$ la valeur obtenue par l'algorithme A avec les décisions $\delta_1^A, \ldots, \delta_m^A$.
- **Facteur de compétitivité** sur une instance I (minimisation) :

$$\rho(A, I) = f(I, A)/f^*(I)$$

- **Facteur de compétitivité dans le pire des cas** : pire facteur de compétitivité sur l'ensemble des instances $\mathcal{I}$

$$\rho(A) = \max_{I \in \mathcal{I}} f(I, A)/f^*(I)$$

- **Algorithme c-compétitif** : Un algorithme A est c-compétitif si $\rho(A) \leq c$

Objectif : obtenir un algorithme en ligne du meilleur facteur de compétitivité possible.

Analyse de compétitivité

Solution de l'exercice "Acheter ou louer"

On considère un coût de B pour acheter et un côut de 1 pour louer. Une requête correspond aux décisions : "acheter ou louer". L'inconnue est limitée ici aux nombres de requêtes m (nombre de jours de ski).

Si $m \geq B$, l'optimum est d'acheter à la première requête (coût B). Si $m \leq B$ l'optimum est de toujours louer (coût m).

Pour évaluer le facteur de compétitivité d'une décision, on introduit le concept d'adversaire. si j'achète à la requête k à combien l'adversaire va fixer m pour me faire perdre le plus d'argent possible ?

Si $1 \leq k \leq B$, le coût est de $(k-1)+B$ et le pire des cas est $m = k$ avec un optimum de k ce qui donne un facteur de compétitivité de $1 + \frac{B-1}{k}$. si $k \geq B$, le coût est aussi de $(k-1)+B$ et le pire des cas est aussi $m = k$ avec un optimum de B et un facteur de compétitivité de $1 + \frac{k-1}{B}$.

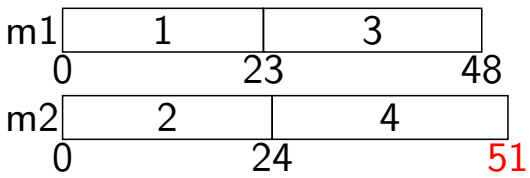
L'algorithme optimal est de choisir la valeur de k qui minimise ces fonctions : $k = B$ avec un facteur de compétitivité de $2 - \frac{1}{B}$. Pour $B = 10$ on obtient $1,9$.

Analyse de compétitivité

Problème d'ordonnancement

Montrer l'algorithme d'ordonnancement OPLUSTOT sur la machine qui se libère le plus tôt pour un modèle d'ordonnancement en ligne de liste sur m machines parallèles est 2-compétitif.

- Calculer une borne inférieure de la durée totale minimale en utilisant les durées des tâches et le nombre de machines.
- Comparer la durée de l'ordonnancement obtenu par OPLUSTOT avec cette borne inférieure.



Exemple d'algorithme en ligne utilisé en pratique

Problème d'allocation de créneaux de décollage

- Gestion des flux de traffic aérien (ATFM, Air Traffic Flow Management)
- Tout vol commercial doit déposer un plan de vol 3 heures avant le décollage, indiquant l'ETO (Estimated Take Over), l'heure de décollage estimée.
- En Europe, C'est le service CFMU (Central Flow Management Unit) d'EUROCONTROL qui attribue les créneaux de décollage (slots) aux avions dans les aéroports avec un système partiellement automatisée : le système CASA (Computer Assisted Slot Allocation).
- CASA attribue à tout vol déclaré un slot (CTOT : Calculated Take Off Time), c'est à dire une heure précise de décollage avec une tolérance de -5mn et de +10mn) via un message, envoyé au plus tôt deux heures avant le mouvement estimé de l'avion nécessaire au départ (EOBT - Estimated Block off time)
- L'algorithme CASA est lancé toutes les minutes. Il utilise la règle FPFS, (first planned first served). Un message de révision de créneau est envoyé au vol en cas de changement.
- l'objectif est de minimiser la somme des retards totaux $\sum_{\text{vols}}$ (CTOT - ETO)

Exemple d'algorithme en ligne utilisé en pratique

Algorithme CASA

Entrées

- Ensemble des vols déclarés $\mathcal{F} = \{1, \ldots, F\}$
- Ensemble des créneaux $\mathcal{T} = \{1, \ldots, T\}$,
- Pour chaque vol $f \in \mathcal{F}$, $EOT_f, \forall f \in \mathcal{F}$ le slot de décollage souhaité

Sorties

- Pour chaque vol $f \in \mathcal{F}$, le slot $CTOT_f$ alloué

Algorithme CASA

1: Tri des vols déclarés $f \in \mathcal{F}$ dans l'ordre croissant des $EOT_f \rightarrow$ liste $\mathcal{L}$
2: **for** $f \in \mathcal{L}$ **do**
3: Prendre le premier slot $t \in \mathcal{T}$ disponible tel que $t \geq EOT_f$ et assigner $CTOT_f \leftarrow t$
4: **end for**

Exemple d'algorithme en ligne utilisé en pratique

Algorithme CASA

Theorem

L'algorithme CASA est optimal pour minimiser la somme des écarts $\sum_{f \in \mathcal{F}} CTOT_f - EOT_f$

Démonstration.

Preuve au tableau □

L'algorithme est en ligne car il est relancé toutes les minutes, avec un ensemble de vols déclarés qui peut changer. Les vols à la date de décollage trop proche sont figés et ne peuvent pas être impactés par la replanification $\Longrightarrow$ perte de la propriété d'optimalité

Randomisation

Algorithme randomisé

- Un algorithme randomisé RA peut être vu comme un ensemble d'algorithmes déterministes $\{A_y\}$ associé à une distribution de probabilité Y.

Exemple

Pour le problème P d'"Acheter ou louer" considérons l'algorithme non déterministe $\tilde{A}_B$ suivant muni d'une distribution de probabilité discrète. Soit A_B l'algorithme déterministe consistant à acheter à la Bème journée de ski :

$$\tilde{A}_B(p) = \begin{cases} A_{B/2} & \text{avec la probabilité } p \\ A_B & \text{avec la probabilité } (1-p) \end{cases}$$

- Concept d'**Adversaire faible** (*Oblivious adversary*): l'adversaire génère l'instance I en connaissant l'algorithme RA, mais sans connaître les résultats d'aucune des décisions aléatoires (l'instance doit être générée à l'avance). L'adversaire paye le coût optimal $f^*(I)$.
- L'adversaire faible correspond à la transposition dans le cas non déterministe du facteur de compétitivité.
- Un algorithme randomisé RA est c-compétitif contre un adversaire faible si, pour toute instance $I = (\sigma_i)_{i=1,\ldots,m}$,

$$\mathbb{E}_Y[f(I, RA_y)] \leq cf^*(I)$$

Exercice

Pour le problème P d'"Acheter ou louer" considérons l'algorithme non déterministe $\tilde{A}_B$ suivant muni d'une distribution de probabilité discrète. Soit A_B l'algorithme déterministe consistant à acheter à la Bème journée de ski :

$$\tilde{A}_B(p) = \begin{cases} A_{B/2} & \text{avec la probabilité } p \\ A_B & \text{avec la probabilité } (1-p) \end{cases}$$

Calculer la probabilité p qui minimise le facteur de compétitivité $\max_{j\in\mathbb{N}} \mathbb{E}[f(j, \tilde{A}_B)(p)]/f^*(j)$ contre tout adversaire faible où j est le nombre de jours de ski (définissant une instance).

Calcul au tableau...

On obtient $p = \frac{2}{5}\left(1 - \frac{1}{5B-4}\right)$ et $\max_I \mathbb{E}[f(I, \tilde{A}_B(p))] = \frac{9}{5} - \frac{1}{B}\left(1 - \frac{B}{25B-20}\right)$, ce qui pour $B = 10$ donne $p \approx 0,391$ et un facteur de compétitivité en espérance d'environ 1,704.

Théorème

- Soit $\rho(R) = \max_{I \in \mathcal{I}} \frac{\mathbb{E}_Y f(I,R)}{f^*(I)}$ le facteur de compétitivité d'un algorithme randomisé R contre tout adversaire faible (L'ensemble des algorithmes randomisés est considéré comme l'ensemble des algorithmes déterministes A_y associés à une distribution de probabilité Y).
- Soit P une distribution de probabilités pour sélectionner une instance I (I est une variable aléatoire suivant la distribution P).
- Soit $\mathbb{E}_P[f(I,A)]$ l'espérance du coût d'un algorithme déterministe $A \in \mathcal{A}$ si l'instance est générée selon P, avec $\mathcal{A}$ l'ensemble des algorithmes en ligne déterministes pour le problème. Soit $\mathbb{E}_P[f^*(I)]$, l'espérance de du coût de l'algorithme déterministe optimal si l'instance est générée selon P.

$$\rho(R) \geq \min_{A \in \mathcal{A}} \frac{\mathbb{E}_P[f(I,A)]}{\mathbb{E}_P[f^*(I)]}$$

On en déduit une technique pour borner inférieurement le facteur de compétitivité $\rho(R)$. Il suffit de choisir une distribution de probabilités particulière P et de trouver une borne inférieure LB telle que $\min_{A \in \mathcal{A}} \frac{\mathbb{E}_P[f(I,A)]}{\mathbb{E}_P[f^*(I)]} \geq LB$. On a alors $\min_{R \in \mathcal{R}} \rho(R) \geq LB$.

Considérons le problème du skieur avec un coût $B = 10$. Considérons l'ensemble des instances I suivant la distribution de probabilité P telle qu'il n'y ait qu'un seul jour de ski avec une probabilité $1/2$ et 20 jours de ski avec une probabilité $1/2$.

L'espérance du côut optimal est $\mathbb{E}_p[f^*(I)] = \frac{1}{2}.1 + \frac{1}{2}.10 = 11/2$.

Soit A_j l'algorithme déterministe qui achète le jour j. On a

$$\mathbb{E}_p[f(I,A_j)] = \begin{cases} 10 & \text{pour } j = 1 \\ \frac{1}{2}.1 + \frac{1}{2}(j-1+10) & \text{pour } j = 2,\ldots,20 \\ \frac{1}{2}.1 + \frac{1}{2}.20 & \text{pour } j > 20 \end{cases} \geq \frac{1}{2} + \frac{11}{2} = 6.$$

On a alors $\frac{\mathbb{E}_p[f(I,A_j)]}{\mathbb{E}_p[f^*(I)]} \geq \frac{12}{11}$. Aucun algorithme en ligne ne peut être c-compétitif avec $c \geq 12/11$.

sources

- **Susanne Albers**. BRICS. Mini course on Competitive Online Algorithms. MPI Saarbrücken. **Aarhus University**. August 27–29 1996. http://www.brics.dk/LS/96/2/
- **Sven O. Krumke**. Online Optimization. OptALI Summer School Auckland, 2011. **Technische Universität KaiserSlautern**.
- **Jim Sukha**. 6.046, Introduction to Algorithms. Recitation Notes on Competitive analysis. **MIT** http://people.csail.mit.edu/sukhaj/6.046/rec9.pdf
- **Luca Trevisan**. CS261 Lecture 17: Online Algorithms. **Standford University**. http://lucatrevisan.wordpress.com/2011/03/09/cs261-lecture-17-online-algorithms/
- **Rudolf Fleischer**, COMP 670K- Topics in Theoretical CS - Online Algorithms - Fall 2000, **Fudan University** http://www.tcs.fudan.edu.cn/rudolf/Courses/Online/Online00/index.html
- **Andrea Ranieri** and **Lorenzo Castelli**. A Market Mechanism to Assign Air Traffic Flow Management Slots. 8th USA/EUROPE ATM 2009 R&D Seminar, Napa Valley, California, USA, 29 June-2 July 2009. http://atmseminarus.org/seminarContent/seminar8/papers/p_086_FP.pdf

Programmation linéaire avec contraintes probabilistes
Contraintes en probabilités séparées

$$\begin{aligned}(\text{PL}) \qquad & \min c_1x_1 + c_2x_2 + \ldots + c_nx_n \\ & a_{1j}x_1 + a_{2j}x_2 + \ldots + a_{nj}x_n \geq b_j \qquad j = 1, \ldots, m \\ & x_i \geq 0 \qquad i = 1, \ldots, n\end{aligned}$$

Programme linéaire avec contraintes probabilistes séparées

$$\begin{aligned}(\text{PLCP}) \qquad & \min c_1x_1 + c_2x_2 + \ldots + c_nx_n \\ & \mathbb{P}\{\tilde{a}_{1j}x_1 + \tilde{a}_{2j}x_2 + \ldots + \tilde{a}_{nj}x_n \geq \tilde{b}_j\} \geq \alpha_j \quad j = 1, \ldots, m \\ & x_i \geq 0 \qquad i = 1, \ldots, n\end{aligned}$$

avec $(\tilde{a}, \tilde{b}) \in \Omega \subseteq \mathbb{R}^{m \times n} \times R^m$ vecteurs aléatoires de distributions connues et $1 - \alpha_j$: risque maximum acceptable pour la contrainte j.

Programmation linéaire avec contraintes probabilistes
Contraintes en probabilités jointes

$$\begin{aligned}(\text{PL}) \qquad & \min c_1x_1 + c_2x_2 + \ldots + c_nx_n \\ & a_{1j}x_1 + a_{2j}x_2 + \ldots + a_{nj}x_n \geq b_j \qquad j = 1, \ldots, m \\ & x_i \geq 0 \qquad i = 1, \ldots, n\end{aligned}$$

Programme linéaire avec contraintes probabilistes jointes

$$\begin{aligned}(\text{PLCP}) \qquad & \min c_1x_1 + c_2x_2 + \ldots + c_nx_n \\ & \mathbb{P}\left\{ \begin{array}{l} \tilde{a}_{1j}x_1 + \tilde{a}_{2j}x_2 + \ldots + \tilde{a}_{nj}x_n \geq \tilde{b}_j \\ j = 1, \ldots, m \end{array} \right\} \geq \alpha \\ & x_i \geq 0 \quad i = 1, \ldots, n\end{aligned}$$

avec $(\tilde{a}, \tilde{b}) \in \Omega \subseteq \mathbb{R}^{m \times n} \times R^m$ vecteurs aléatoires de distributions connues et $1 - \alpha$: risque global maximum acceptable.

Programmation linéaire avec contraintes probabilistes

Ensemble des solutions

$C(\alpha) = \{x \in \mathbb{R}^n : p(x) \geq \alpha\}$ avec deux représentations alternatives pour la fonction de confiance $p(x)$

- $p(x) = \mathbb{P}\left\{x \in S(\tilde{a}, \tilde{b})\right\}$ où pour $(a, b) \in \Omega$,

$$S(a,b) = \left\{ x \in \mathbb{R}^n \;\middle|\; \begin{array}{l} a_{1j}x_1 + a_{2j}x_2 + \ldots + a_{nj}x_n \geq b_j \\ j = 1, \ldots, m \end{array} \right\}$$

  $=$ ensemble des x réalisables si la réalisation des variables aléatoires est (a, b)

- $p(x) = \mathbb{P}\left\{(\tilde{a}, \tilde{b}) \in K(x)\right\}$ où pour $x \in \mathbb{R}^n$,

$$K(x) = \left\{ (a,b) \in \Omega \;\middle|\; \begin{array}{l} a_{1j}x_1 + a_{2j}x_2 + \ldots + a_{nj}x_n \geq b_j \\ j = 1, \ldots, m \end{array} \right\}$$

  $=$ ensemble des réalisations des variables aléatoires favorables à x

Programmation linéaire avec contraintes probabilistes

Exemple le plus simple

$$\mathbb{P}[x \geq \tilde{b}] \geq \alpha$$

Ensemble des x réalisables pour la réalisation b : $S(b) = [b, \infty)$

Ensemble des réalisations b favorables pour un x donné : $K(x) = (-\infty, x]$

Fonction de confiance
$p(x) = \mathbb{P}[x \in S(\tilde{b})] = \mathbb{P}[\tilde{b} \in K(x)] = P[x \geq \tilde{b}] = F(x)$ où $F(x)$ est la fonction de répartition de la variable aléatoire $\tilde{b}$

Ensemble réalisable $C(\alpha) = \{x \in \mathbb{R}^n : p(x) \geq \alpha\} = [F^{-1}(\alpha), \infty)$, s'il est possible d'inverser la fonction de répartition F

Alors, l'inégalité $P[x \geq \tilde{b}] \geq \alpha$ a un équivalent déterministe

$$x \geq F^{-1}(\alpha)$$

Programmation linéaire avec contraintes probabilistes

Généralisation : contraintes probabilistes séparées avec second membre aléatoire

Le programme linéaire avec contraintes en probabilités séparées et où seul le second membre est aléatoire (vecteur $\tilde{b}$)

$$\mathbb{P}[a_{1j}x_1 + a_{2j}x_2 + \ldots + a_{2n}x_n \geq \tilde{b}_j] \geq \alpha \quad j = 1, \ldots, m$$

a un équivalent déterministe

$$a_{1j}x_1 + a_{2j}x_2 + \ldots + a_{2n}x_n \geq F^{-1}(\alpha) \quad j = 1, \ldots, m$$

Le problème est de calculer $F^{-1}(\alpha)$

Programmation linéaire avec contraintes probabilistes

Exemple à une seule composante aléatoire dans les coefficients

Soit une contrainte où seul un coefficient des variables de décision est aléatoire:

$$\mathbb{P}[\tilde{a}x_1 + 2x_2 \geq 8] \geq \alpha \text{ avec } p(x) = P[\tilde{a}x_1 + 2x_2 \geq 8]$$

Si $x_1 > 0$ on obtient $p(x) = P[\tilde{a} \geq \frac{(8-2x_2)}{x_1}] = 1 - P[\tilde{a} \leq \frac{(8-2x_2)}{x_1}]$

Si $x_1 < 0$ on obtient $p(x) = P[\tilde{a} \leq \frac{(8-2x_2)}{x_1}]$

Si $x_1 = 0$, la contrainte devient déterministe $x_2 \geq 4$. d'où

$$p(x) = \mathbb{P}[\tilde{a}x_1 + 2x_2 \geq 8] = \begin{cases} 1 - F(\frac{8-2x_2}{x_1}) & x_1 > 0 \\ 0 & x_1 = 0 \text{ et } x_2 < 4 \\ 1 & x_1 = 0 \text{ et } x_2 \geq 4 \\ F(\frac{8-2x_2}{x_1}) & x_1 \leq 0 \end{cases}$$

$C(\alpha) = \{x : x_1 = 0, x_2 \geq 4\} \cup \{x : x_1 > 0, F^{-1}(1-\alpha)x_1 + 2x_2 \geq 8\} \cup \{x : x_1 \leq 0, F^{-1}(\alpha)x_1 + 2x_2 \geq 8\}$

Exercice : calculer $C(\alpha)$ avec $\tilde{a} \sim \mathcal{U}[3, 4]$, $\alpha = 0,4$, $\alpha = 0,5$ et $\alpha = 0,7$.

Programmation linéaire avec contraintes probabilistes

Exemple avec scénarios associés à des probabilités discrètes

Soit $x \in \mathbb{R}^2$. On considère les scénarios :

$$\tilde{a} = (\tilde{a}_1, \tilde{a}_2) = \begin{cases} (3,0) & \text{avec prob. } 1/7 \\ (0,3) & \text{avec prob. } 2/7 \\ (1,1) & \text{avec prob. } 4/7 \end{cases}$$

et la contrainte probabiliste $\mathbb{P}[\tilde{a}_1 x_1 + \tilde{a}_2 x_2 \leq 1] \geq \alpha$

$$S(a) = \begin{cases} \{x \in \mathbb{R}^2 : x_1 \leq 1/3\}; & \text{pour } a = (3,0) \\ \{x \in \mathbb{R}^2 : x_2 \leq 1/3\}; & \text{pour } a = (0,3) \\ \{x \in \mathbb{R}^2 : x_1 + x_2 \leq 1\}; & \text{pour } a = (1,1) \end{cases}$$

Exercice : calculer $C(\alpha)$ pour $\alpha = 6/7$

Programmation linéaire avec contraintes probabilistes

Convexité

Autre expression de $C(\alpha)$. Soit K_α l'ensemble des ensembles K de réalisation de $(\tilde{a}, \tilde{b})$ tels que $\mathbb{P}[(\tilde{a}, \tilde{b}) \in K] \geq \alpha$. Alors

$$C(\alpha) = \cup_{K \in K_\alpha} \cap_{(a,b) \in K} S(a,b)$$

$S(a,b)$ est un ensemble convexe donc $\cap_{(a,b) \in K} S(a,b)$ aussi. Par contre $C(\alpha)$ en tant qu'union d'ensembles convexes n'est en général pas convexe.

- Pour les cas où $C(\alpha)$ est convexe, il existe des méthodes efficaces pour trouver un point ou optimiser dans $C(\alpha)$.
- $C(\alpha) = \{x \in \mathbb{R}^n : p(x) \geq \alpha\}$ est convexe si $p(x)$ est quasi-concave

Cas où $C(\alpha)$ est convexe. Les résultats sont valables pour des contraintes en probabilité séparées.

- Seuls les membres de droite $\tilde{b}$ sont aléatoires avec une fonction de répartition F inversible, on obtient un programme linéaire

$$a_{1j}x_1 + a_{2j}x_2 + \ldots + a_{2n}x_n \geq F^{-1}(\alpha) \quad j = 1, \ldots, m$$

- Seul un coefficient $\tilde{a}_{ij}$ est aléatoire. Convexe si $\alpha \geq 1/2$.
  $C(\alpha) = \{x : x_1 = 0, x_2 \geq 4\} \cup \{x : x_1 > 0, F^{-1}(1-\alpha)x_1 + 2x_2 \geq 8\} \cup \{x : x_1 \leq 0, F^{-1}(\alpha)x_1 + 2x_2 \geq 8\}$ convexe si
  $-F^{-1}(\alpha) \leq -F^{-1}(1-\alpha) \Leftrightarrow \alpha \geq 1/2$

$\mathbb{P}\{\tilde{a}_{1j}x_1 + \tilde{a}_{2j}x_2 + \ldots + \tilde{a}_{nj}x_n \geq b_j\} \geq \alpha_j$
avec b fixé et $\tilde{a} \sim \mathcal{N}_n(\mu, \Sigma)$ avec $\mu \in \mathbb{R}^n$ et $\Sigma \in \mathbb{R}^{n \times n}$ (matrice $n \times n$)

- $\tilde{a}^T x - b \sim \mathcal{N}(\mu^T x - b, \sigma^2(x))$ avec $\sigma^2(x) = x^T \Sigma x$.
- Normalisation. On pose $\tilde{u} = \frac{\tilde{a}x - \mu^T x}{\sigma(x)} \sim \mathcal{N}(0,1)$ de fonction de répartition Φ.
- donc $\mathbb{P}[\tilde{a}x \geq b] = P[\tilde{u} \geq \frac{b - \mu^T x}{\sigma(x)}] = \Phi(\frac{\mu x - b}{\sigma(x)})$
- $C(\alpha) = x \in \mathbb{R}^n : \mu^T x \geq b + \Phi^{-1}(\alpha)\sigma(x)$
- $\alpha \geq 1/2 \Rightarrow \Phi^{-1}(\alpha) \geq 0 \Rightarrow$
- $C(\alpha)$ définit un programme linéaire si $\Phi^{-1}(\alpha) = 0$
- si $\Phi^{-1}(\alpha) > 0$, on obtient $\sigma(x) \leq \frac{1}{\Phi^{-1}(\alpha)}\mu^T x - \frac{b}{\Phi^{-1}(\alpha)}$ avec $\sigma(x) = \sqrt{x^T \Sigma x} = ||\Sigma^{\frac{1}{2}} x||_2$. $C(\alpha)$ est convexe et définit un programme sur le cône du second ordre.

Programmation linéaire avec contraintes probabilistes
Optimisation de portefeuille

- Ensemble d'actifs financiers avec des revenus (aléatoires) $\tilde{a}_1, \tilde{a}_2, \ldots, \tilde{a}_n$
- $\tilde{a} \sim \mathcal{N}(\mu, \Sigma)$
- On doit déterminer un pourcentage d'investissement dans chacun des actifs.
- Ecrire un programme mathématique qui permet d'avoir un gain attendu T avec une certaine probabilité α.

Programmation linéaire avec contraintes probabilistes
Modèle de programmation linéaire en nombres entiers pour la distribution discrète

- Scénarios (a^k, b^k), $k \in \{1, \ldots, K\}$ de probabilités p_k.
- On introduit une variable binaire $\delta_k \in \{0, 1\}$ telle que $\delta_k = 1 \Leftrightarrow x \in S(k)$
- Cela s'exprime par les contraintes $a^k x + M(1 - \delta_k) \geq \tilde{b}^k$, $k = 1, \ldots, K$
- On impose ensuite $\sum_{k=1}^{K} p_k \delta_k \geq \alpha$

Modèles avec recours

Programmation stochastique à deux niveaux

- On a considéré que toutes les variables doivent être fixées avant que les paramètres aléatoires soient révélés.
- Dans certains cas seules certaines variables (dites de premier niveau) sont soumises à cette règle.
- Les autres variables (de deuxième niveau ou de **recours**) permettent d'ajuster la solution une fois les paramètres aléatoires révélés.
- Ces variables de recours permettent en général d'assurer la faisabilité des solutions de premier niveau

Modèles avec recours

Programmation linéaire stochastique à deux niveaux

- x vecteur de variables de décision de premier niveau
- y vecteur de variables de décision de second niveau (recours)
- c vecteur coût des variables de premier niveau
- $\tilde{q}$ vecteur coût des variables de second niveau (stochastique)
- A, b matrice des contraintes et second membre du premier niveau
- $\tilde{T}, \tilde{W}, \tilde{d}$ matrices des contraintes et second membre du deuxième niveau (stochastiques)

Premier niveau

$$\min c^T x + E[h(x, \tilde{q}, \tilde{T}, \tilde{W}, \tilde{d})]$$
$$Ax \geq b$$
$$x \geq 0$$

Deuxième niveau

$$h(x, c, T, W, d) = \min q^T y$$
$$Tx + Wy \geq d$$
$$y \geq 0$$

Optimisation stochastique à deux niveaux

Exercice

Au début de chaque mois une entreprise doit décider de sa production en fonction d'une demande aléatoire dont la distribution est connue. La variable x_t décide combien produire au mois t en fonction du paramètre aléatoire $\tilde{d}_t$ de demande au mois t et du stock disponible en début de période y_{t-1}. Si la demande révélée est inférieure à x_t, un recours consiste à stocker le surplus (moyennant un coût de stockage h_t). Si elle est supérieure, un recours consiste à acheter les produits manquant (moyennant un coût d'approvisionnement b_t). L'objectif est la minimisation de l'espérance des coûts totaux de production, de stockage et de rupture.

Ecrire un programme linéaire permettant de modéliser le problème de décision sur le premier mois $t = 1$, en considérant un stock initial y_0, un coût de stockage h_1, un coût de réapprovisionnement b_1 et en considérant que la demande $\tilde{d}_1$ suit une loi discrète sous forme de scénarios S_t, chaque scénario $s \in S_t$ définissant une demande d_t^s avec une probabilité p_t^s.:

Application numérique : $b_1 = 3$, $h_1 = 2$, $|S_1| = 2$, scénario 1 : $d_1^1 = 500$, $p_1^1 = 0.6$, scénario 2 : $d_1^2 = 700$, $p = 0.4$.

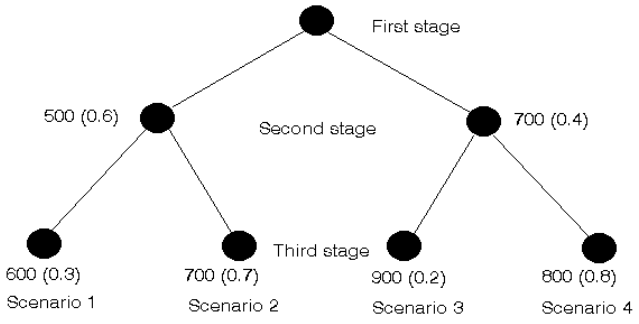
Optimisation stochastique multi-étapes

- Il s'agit d'une généralisation de l'optimisation stochastique à deux niveaux.
- Les paramètres aléatoires sont progressivement révélés dans une série d'étapes. A chaque étape e
  1. les variables de l'étaple e sont fixées (action)
  2. Les paramètres aléatoires de l'étape e sont révélés (observation)
  3. Les variables de recours de l'étape e sont fixées (réaction)
- Le processus se répète jusqu'à l'étape finale. L'objectif est la minimisation de l'espérance du coût total.

Optimisation stochastique Multi-étapes

Exercice

On considère un cas particulier à trois étapes avec un coût de production unitaire de 2, un coût unitaire d'approvisionnement de 3 et un coût de stockage nul. La demande aléatoire $\tilde{d}_t$ est définie par 4 scénarios décrits dans la figure ci-dessous. Formuler le problème de minimisation de l'espérance du coût total.



A two-level (three-stage) binary scenario tree
Numbers are demand(probability)

http://http://people.brunel.ac.uk/~mastjjb/jeb/or/sp.html

Modèles avec recours

Programmation linéaire en nombres entiers stochastique à deux niveaux

- x vecteur de variables de décision de premier niveau (p variables entières et g variables fractionnaires)
- y vecteur de variables de décision de second niveau (recours) (p' variables entières et g' variables fractionnaires)

Premier niveau

$$\min c^T x + E[h(x, \tilde{q}, \tilde{T}, \tilde{W}, \tilde{d})]$$
$$Ax \geq b$$
$$x_1, \ldots, x_p \in \mathbb{N}$$
$$x_{p+1}, \ldots, x_{p+g} \geq 0$$

Deuxième niveau

$$h(x, c, T, W, d) = \min q^T y$$
$$Tx + Wy \geq d$$
$$y_1, \ldots, y_{p'} \in \mathbb{N}$$
$$y_{p'+1}, \ldots, y_{p'+g'} \geq 0$$

Modèles avec recours

Programmation linéaire en nombres entiers stochastique à deux niveaux

Pour les probabilité discrètes, décrites par des scénarios $s \in S$ associant à des probabilités p_s les paramètres T_s, q_s, W_s, d_s, on peut obtenir un équivalent déterministe au problème stochastique.

$$\min c^T x + \sum_{s \in S} p_s {q_s}^T y_s$$

$$Ax \geq b$$

$$T_s x + W_s y_s \geq d_s, \quad \forall s \in S$$

$$x_1, \ldots, x_p \in \mathbb{N}$$

$$x_{p+1}, \ldots, x_{p+g} \geq 0$$

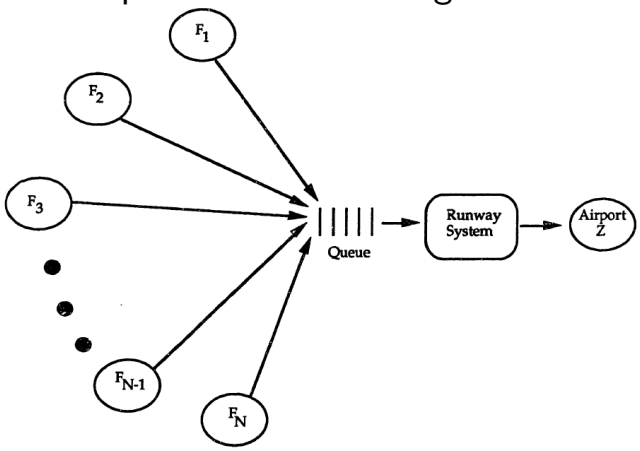
$$y_{s1}, \ldots, y_{sp'} \in \mathbb{N}, \quad \forall s \in S$$

$$y_{s,p'+1}, \ldots, y_{s,p'+g'} \geq 0, \quad \forall s \in S$$

Exemple de programme stochastique multi-étapes

Stratégies de maintien au sols d'aéronefs en présence d'incertitudes

Soit un ensemble de N vols qui convergent vers un aéroport Z. La capacité de l'aéroport étant limitée, les vols qui ne peuvent pas atterrir en cas de saturation peuvent être mis en file d'attente en vol mais il peut être préférable de retarder leur départ. C'est la stratégie de maintien volontaire au sol.



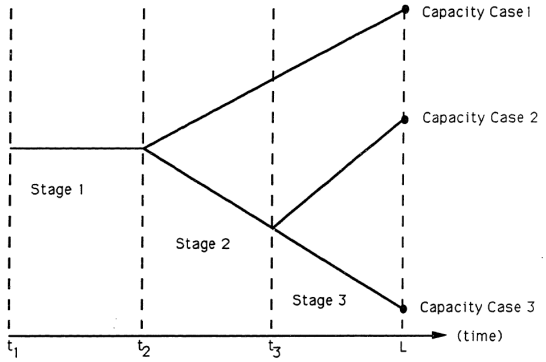
source [Richetta 1991]

- c_S cout de maintien au sol par unité de temps
- c_V cout de maintien en file d'attente en vol par unité de temps
- $\mathcal{T} = \{1, \ldots, L\}$ ens. des périodes de temps
- $\mathcal{S} = \{1, \ldots, S\}$ ens. des scénarios
- p_s probabilité du scénario $s \in \mathcal{S}$
- K_{ts} nombre d'atterrissages maximum dans le scénario $s \in \mathcal{S}$ pour la période $t \in \mathcal{T}$

Exemple de programme stochastique multi-étapes

Stratégies de maintien au sols d'aéronefs en présence d'incertitudes

Exemple de trois scénarios, définis en trois étapes

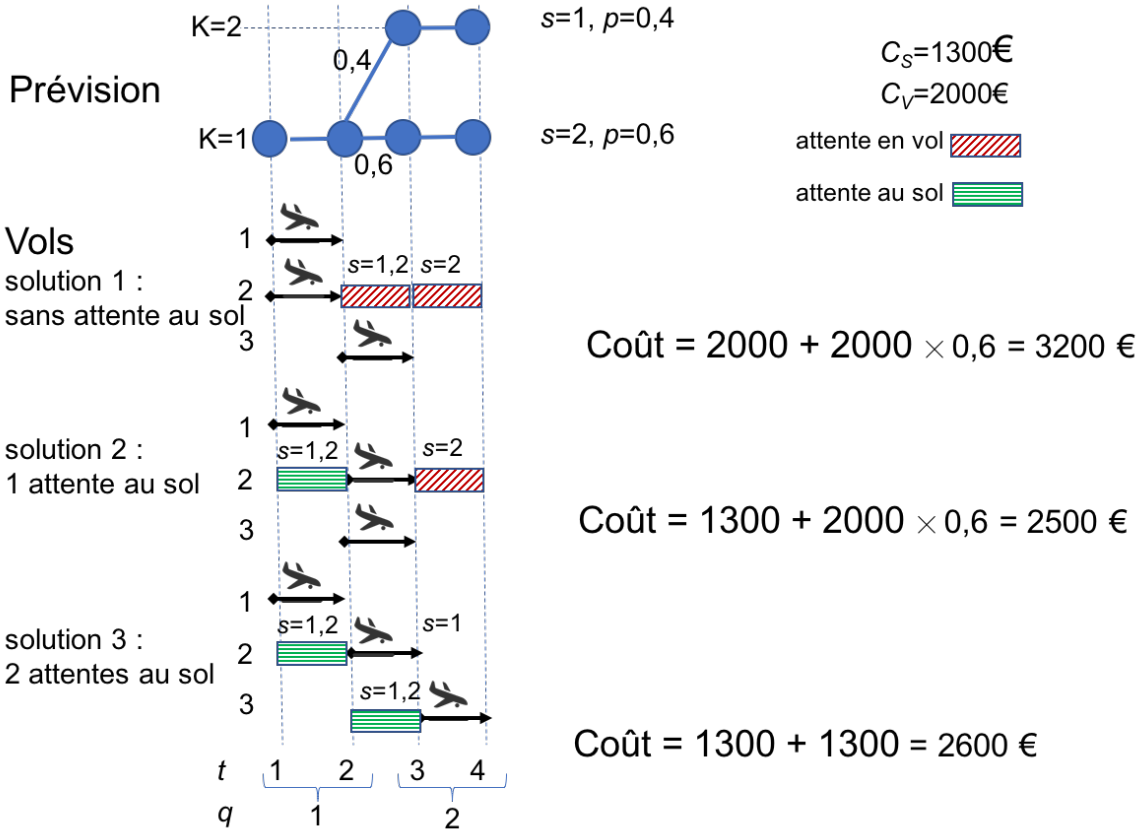


source [Richetta 1991]

- $\mathcal{Q} = \{1, \ldots, Q\}$ nombre d'étapes
- $\mathcal{P}_q$ partition des scénarios à l'étape q (ex: $\mathcal{P}_1 = \mathcal{S} = \{1,2,3\}$, $\mathcal{P}_2 = \{1\},\{2,3\}$, $\mathcal{P}_3 = \{1\},\{2\},\{3\}$)
- $t_q \in \mathcal{T}$ début de l'étape $q \in \mathcal{Q}$ (ex à t_2 soit on suit à coup sur la prévision du scénario 1, soit on a une incertitude entre la prévision des scénarios 2 et 3)
- N_{qt} nombre d'avions décollant dans l'étape $q \in \mathcal{Q}$ et devant atterrir à la période $t \in \mathcal{T}$ (avec $t > t_q$)

Exemple de programme stochastique multi-étapes

Stratégies de maintien au sols d'aéronefs en présence d'incertitudes



Exemple de programme stochastique multi-étapes

Stratégies de maintien au sols d'aéronefs en présence d'incertitudes

Variables de décision

- $X_{qsij} \in \mathbb{N}$ nombre d'avions décollant dans l'étape $q \in \mathcal{Q}$, devant initialement atterrir à la période $i \in \mathcal{T}$ et décalés à la période $j \in \mathcal{T}$, $j \geq i$ dans le scénario $s \in \mathcal{S}$ (ils ont donc été maintenus au sol $j - i$ périodes).
- $W_{si} \in \mathbb{N}$ nombre d'avions dans la file d'attente en vol dans la période $i \in \mathcal{T}$ pour le scénario $s \in \mathcal{S}$

Remarques

- Les décisions prises à une étape q ne peuvent pas être remises en cause à l'étape $q + 1$
- On suppose qu'au début de l'étape q, la capacité de la période est connue (on sait donc dans quelle "branche" de l'arbre des scenario on se situe jusqu'à ce point.).
- A une étape donnée, on ne peut par contre pas anticiper les scenario révélés aux étapes futures

Exemple de programme stochastique multi-étapes

Stratégies de maintien au sols d'aéronefs en présence d'incertitudes

Exercice : Ecrire le programme linéaire à trois niveaux avec les scénarios de l'exemple de [Richetta 1991]

Sources

- M. Khouja. The single-period (newsvendor) problem : literature review and suggestions for future research. Omega 27. p. 537–553, 1999
- J.-F. Hêche, T. M. Liebling. D. de Werra. Recherche Opérationnelle pour Ingénieurs II. Presses Polytecniques et Universitaires Romandes. 2003
- A. Shapiro, D. Dentcheva, A. Ruszczyński. Lectures on Stochastic Programming, Modeling and Theory. MPS-SIAM Series on Optimization, 2009.
- Guillermo Gallego. Guillermo Gallego. "IEOR 4000 Production Management Lecture 7". Columbia University, 6 april 1995. `http://www.columbia.edu/~gmg2/4000/pdf/lect_07.pdf`
- Maarten van der Vlerk. Stochastic Programming LNMB course SP, spring 2011. University of Groningen. `http://www.rug.nl/feb/MHvanderVlerk/LNMBcourseSP`
- Octavio Richetta, Ground Holding Strategies for Air Traffic Control Under Uncertainty. PhD thesis, MIT, 1991

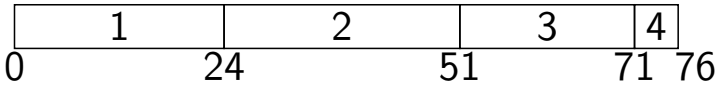
- Il n'est pas toujours facile ni même possible d'obtenir les lois de probabilité des paramètres incertains, encore moins des lois de probabilités indépendantes.
- En pratique, les décideurs peuvent vouloir être très prudent et demander que la décision soit *robuste*, i.e. qu'elle soit la meilleure (ou moins mauvaise) possible si le pire scénario se produit, ce que n'assure pas une optimisation de l'espérance de la fonction objectif.

Exemple : Retour sur le problème d'ordonnancement

$$\tilde{p}_1 \sim \mathcal{U}[23,24] \quad \tilde{p}_2 \sim \mathcal{U}[21,27] \quad \tilde{p}_3 \sim \mathcal{U}[20,29] \quad \tilde{p}_4 \sim \mathcal{U}[5,45]$$

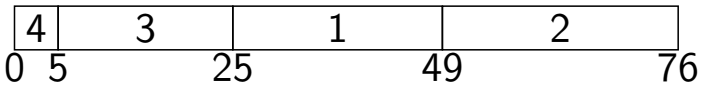
La solution qui minimise l'espérance du critère de somme des dates de fin consiste à ordonnancer les tâches selon l'ordre de la plus petite durée espérée, soit ici $1-2-3-4$ car $E[p_1]=23,5$, $E[p_2]=24$, $E[p_3]=24,5$, $E[p_4]=25$ avec $E[\sum_{j\in J} C_i]=240$.

Exemple : Retour sur le problème d'ordonnancement (suite)
Considérons la réalisation $p_1=24$, $p_2=27$, $p_3=20$, $p_4=5$. En appliquant la solution optimale du problème d'ordonnancement stochastique, on obtient la solution ::



$\sum C_i = 222$

Considérons maintenant la solution optimale pour cette réalisation (ordre des plus petites durées) :



$\sum C_i = 155$

La solution "optimale' ' stochastique excède de 67 unités de temps ou de $43,2\%$ la solution optimale pour la réalisation considérée !

Optimisation Robuste

Critère minimax

- Principe : obtenir des garanties **dans le pire des cas** pour une modélisation raisonnable des incertitudes (sous la forme de scénarios)

Définitions préalables

- S : ensemble (continu ou discret) de scénarios.
- D^s : réalisation des paramètres du problème pour un scénario donné $s \in S$.
- X^s : ensemble des solutions réalisables pour un scénario donné $s \in S$
- $f : X \times S$ fonction objectif : $f(x, s)$ valeur de l'objectif pour le scénario $s \in S$ et la solution $x \in X^s$

Problème d'optimisation robuste (critère minimax)

Trouver $x^* = \text{argmin}_{x \in \cap_{s \in S} X^s} \max_{s \in S} f(x, s)$

Optimisation Robuste

Critère minimax : application au problème d'ordonnancement

On remplace la modélisation des incertitudes sous forme de loi uniforme par les intervalles correspondant à chaque loi :

$$p_1 \in [23, 24] \quad p_2 \in [21, 27] \quad p_3 \in [20, 29] \quad p_4 \in [5, 45]$$

Pour toute séquence de tâche le pire scénario pour la minimisation des dates de fin est clairement celui qui donne la durée maximale à chaque tâche.
Pour ce scénario, la séquence optimale est $1 - 2 - 3 - 4$ selon la règle des plus petites durées. C'est donc la séquence optimale pour le problème minimax. On retrouve dans ce cas la même solution que pour le cas stochastique.

Optimisation Robuste

Critère minimax regret

Principe : Lorsque la solution appliquée est évaluée a posteriori (après la réalisation des données) un décideur peut être intéressé par la minimisation de l'écart dans le pire des cas entre la valeur de la solution qu'il applique et la valeur qu'il aurait pu obtenir s'il avait connu le scénario au départ. C'est le concept de regret $\rho(x, s)$ d'une solution x pour le scénario s

$$\rho(x, s) = f(x, s) - \min_{y \in X^s} f(y, s)$$

Problème d'optimisation robuste (critère minimax regret absolu)

Trouver $x^* = \text{argmin}_{x \in \cap_{s \in S} X^s} \max_{s \in S} \rho(x, s)$

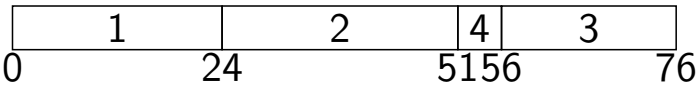
Problème d'optimisation robuste (critère minimax regret relatif)

Trouver $x^* = \text{argmin}_{x \in \cap_{s \in S} X^s} \max_{s \in S} \left(\frac{\rho(x,s)}{\min_{y \in X^s} f(y,s)} \right)$

Optimisation Robuste

Critère minimax regret : application au problème d'ordonnancement

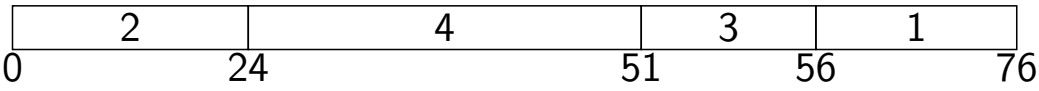
Revenons au scénario $p_1 = 24$, $p_2 = 27$, $p_3 = 20$, $p_4 = 5$, on obtient en appliquant la séquence $1 - 2 - 4 - 3$:



$\sum C_i = 207$

Le regret absolu de cette solution pour ce scénario est de 207-155=52, à comparer avec le regret absolu de la solution optimale stochastique (ou minimax) égal à 62.

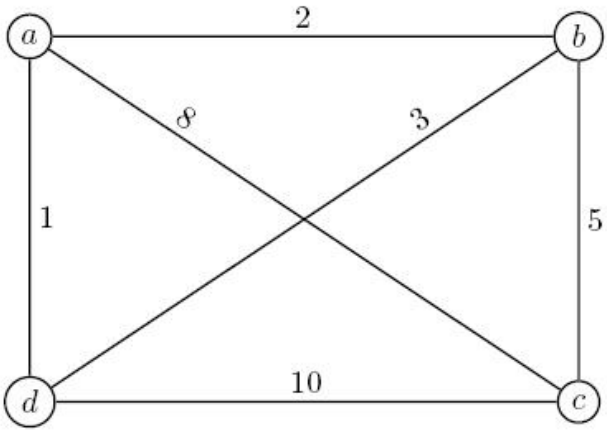
Considérons maintenant le scénario $p_1 = 23$, $p_2 = 27$, $p_3 = 20$, $p_4 = 45$, on obtient en appliquant la séquence $2 - 4 - 3 - 1$



$\sum C_i = 306$

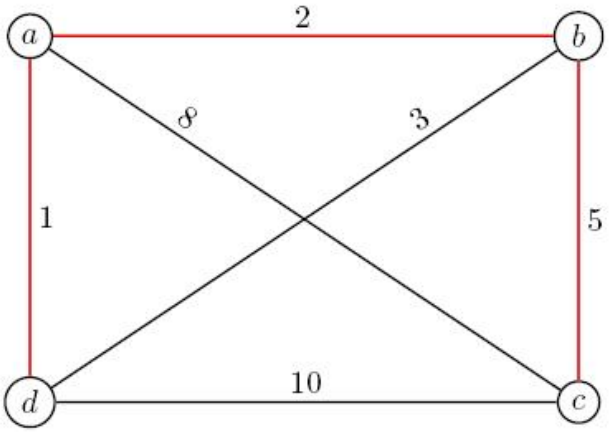
La solution optimale pour ce scénario est $1 - 3 - 2 - 4$ de valeur 248, donnant un regret relatif de $(306 - 248)/248 = 23.4\%$, à comparer avec le regret relatif de la solution optimale stochastique (ou minimax) égal à 43.2%.

Contexte : Conception d'un réseau entre 4 bureaux à moindre coût.



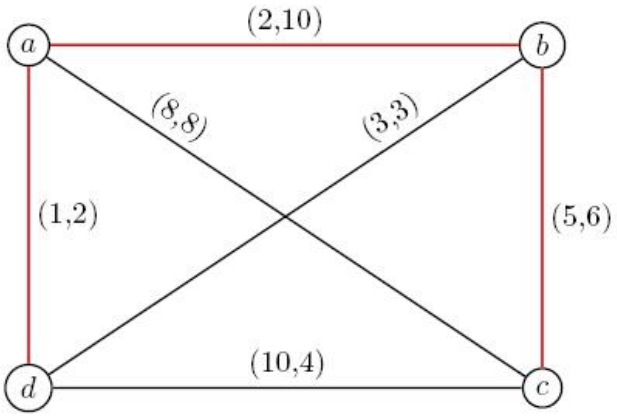
En RO-AD : Déterminer un arbre couvrant (un seul chemin entre tout couple de sommets) de poid minimum.

Contexte : Conception d'un réseau à moindre coût.



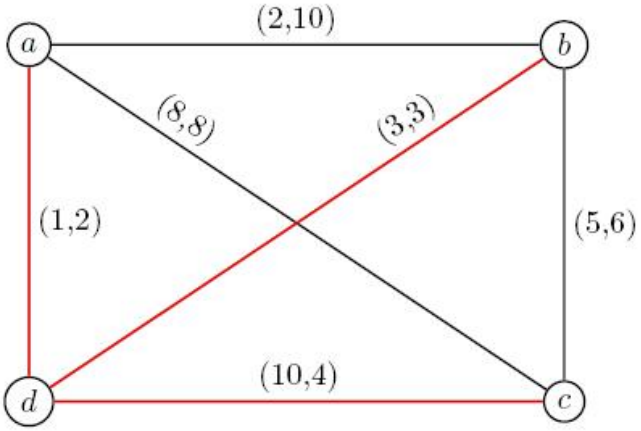
Solution optimale : $\{ad, ab, bc\}$ de coût 8.

Contexte : Conception d'un réseau à moindre coût.



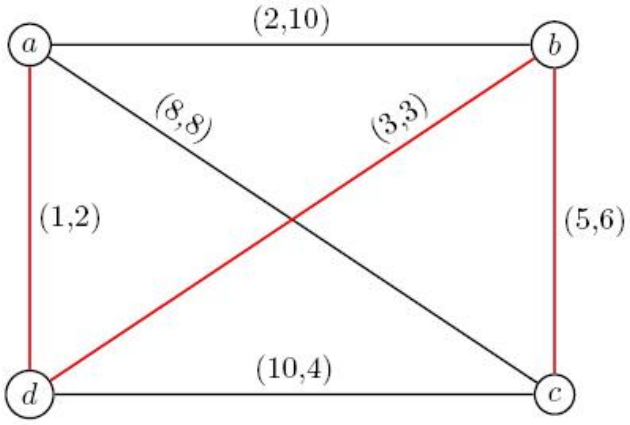
Solution optimale pour scenario 1 : $\{ad, ab, bc\}$ (8,18)

Contexte: Conception d'un réseau à moindre coût.



Solution optimale pour scenario 1 : $\{ad, ab, bc\}$ (8,18)
Solution optimale pour scenario 2 : $\{ad, bd, cd\}$ (14,9)

Contexte : Conception d'un réseau à moindre coût.



Solution opt. pour scenario 1 : $\{ad, ab, bc\}$ (8,18) max =18, Rmax=9
Solution opt. pour scenario 2 : $\{ad, bd, cd\}$ (14,9) max =14, Rmax=6
min-max (regret) ST : $\{ad, bc, bd\}$ (9,11) max =11, Rmax=2

Les approches d'optimisation robuste se distinguent par les différentes modélisation des incertitudes.

- Les scénarios sont donnés explicitement, chaque scénario donnant les valeurs des paramètres incertains pour ce scénario.
- Chaque paramètre incertain et donné sous la forme d'un intervalle de valeurs possibles $[a, b]$ et toutes les combinaisons des valeurs des paramètres sont également possibles.
- Chaque paramètre incertain et donné sous la forme d'une valeur nominale v et d'un écart possible ϵ, mais on suppose qu'au plus k paramètres peuvent s'écarter de leur valeur nominale.
- Autres modélisations (incertitudes ellipsoidales, polyhédrales, ...)

Optimiser pour le pire des scénario est une approche "conservative" (le coût des solutions peut être très élévé). Certaines des modélisations ci-dessus cherchent à réduire le "prix de la robustesse" en restreignant l'espace des scénarios [Bersimas, Sim 2004]

Optimisation Robuste

Exercice : Le problème du marchand de journaux robuste

Un marchand de journaux souhaite déterminer le nombre d'exemplaires Q d'un journal qu'il doit commander. Il ne connait pas la demande précise mais sait qu'elle est comprise entre deux valeurs $[\underline{d}, \overline{d}]$. Chaque exemplaire commandé lui coûte v €. Chaque exemplaire vendu lui rapporte p €. Chaque exemplaire retourné comme invendu lui rapporte g €. Il paye un coût B de perte potentielle de clientèle pour chaque demande non satisfaite.

Calculer l'expression du profit en fonction de la demande d et de la quantité commandée Q.
Résoudre les problèmes d'optimisation robuste avec les critères maximin, maximin regret absolu et maximin regret relatif.
Résolu au cours 1

Ordonnancement robuste avec scénarios explicites - problème minimax

Définition du problème

- On doit ordonnancer un ensemble de tâches T sur une machine. Les tâches sont soumises à des contraintes de précédence. On note $i \prec j$ si j ne peut commencer avant la fin de i. Les tâches sans prédécesseur sont toutes disponibles à la date $t = 0$. Les tâches ont des durées et des dates de livraison incertaines définis par un ensemble S de scénarios tels que p_i^s et d_i^s désignent respectivement la durée et la date de livraison de la tâche i dans le scénario s.
- Soit Σ, l'ensemble des séquences de tâches qui respectent les contraintes de précédence. On souhaite trouver une séquence de tâches $\sigma \in \Sigma$ qui minimise le plus grand retard ($\sigma(i)$ tâche en position i, $\pi(i)$ position de la tâche i).

$$\min_{\sigma \in \Sigma} \max_{s \in S} L_{\max}(\sigma, s) \quad \text{avec} \quad L_{\max}(\sigma, s) = \max_{i \in T} \left(C_i(\sigma, s) - d_i^s\right)$$

$$\text{et } C_i(\sigma, s) = \sum_{j=\sigma(1)}^{\sigma(\pi(i)-1)} p_j^s$$

Ordonnancement robuste avec scénarios explicites - problème minimax

Algorithme de Lawler pour le problème sans incertitude

Un seul scénario : p_i, d_i déterministes

- Remarque : on n'a pas intérêt à insérer de temps d'inactivité sur la machine $\rightarrow$ date de fin de la dernière tâche $= \sum_{i \in T} p_i$.
- Algorithme de Lawler
  Soit $f_i(t) = t - d_i$ et J un ensemble de tâches initialisé aux tâches sans successeur. Q ensemble des tâches ordonnancées.
  - poser $t \leftarrow \sum_{i \in T} p_i$
  - Tant que $J \neq \emptyset$ faire
    - Sélectionner la tâche $j \in J$ de plus petit coût:

$$j = \operatorname*{argmin}_{i \in J} f_i(t)$$

    - Ordonnancer j de façon à ce que $C_j = t$ et poser $t \leftarrow t - p_j$
    - enlever j de J et l'ajouter à Q.
    - Ajouter à J les tâches de $T \setminus Q$ dont tous les successeurs sont dans Q

Ordonnancement robuste avec scénarios explicites - problème minimax

Algorithme pour le problème avec incertitude sur les dates de livraison

p_i (déterministes), $d_i^s, s \in S$ (incertains)

- On définit un pire scénario "artificiel" $d_i^w = \max_{s \in S} d_i^s$, $\forall i \in T$

Théorème (Aloulou et Della Croce (2008))

Appliquer l'algorithme de Lawler au scénario artificiel donne la séquence qui minimise le plus grand retard dans le pire des cas.

Preuve au tableau

Ordonnancement robuste avec scénarios explicites - problème minimax

Algorithme pour le problème avec incertitude sur les durées et les dates de livraison

p_i^s, $d_i^s, s \in S$ (incertains)

- On définit l'algorithme de Lawler "minmax"
- A chaque étape de l'algorithme sélectionner la tâche $j \in J$ qui minimise $\max_{s \in S} f_j^s(\sum_{i \in T \setminus Q} p_i^s)$

Théorème (Aloulou et Della Croce (2008))

Appliquer l'algorithme de Lawler minmax donne la séquence qui minimise le plus grand retard dans le pire des cas.

Preuve au tableau

Ordonnancement robuste avec avec intervalles de durées - problème minimax regret absolu

Définition du problème (1)

- On doit ordonnancer un ensemble de tâches T sur une machine. Les tâches ont des durées incertaines représentées par des intervalles $p_i \in [\underline{p}_i, \overline{p}_i]$. On considère la fonction objectif donnée par la somme des dates de fin des tâches.
- On souhaite trouver une séquence de tâches $\sigma \in \Sigma$ qui minimise le plus grand regret absolu.
- Soit $\sigma(i)$ la tâche en position i et $\pi(i)$ la position de la tâche i dans une séquence. Soit $\sigma_s^*(i)$ la tâche à la position i dans une solution optimale pour le scénario s. Soit $\pi_s^*(i)$ la position de la tâche i dans cette séquence optimale pour le scénario s.
- On remarque que la somme des dates de fin d'une séquence (σ, π) donnée pour un scénario s peut s'écrire

$$\sum_{i \in T} C_i(\sigma, s) = \sum_{i \in T} \pi(i) p_i^s$$

Ordonnancement robuste avec intervalles de durées - problème minimax regret absolu

Définition du problème (2)

- Le regret d'une solution (σ, π) par rapport à une solution optimale (σ_s^*, π_s^*) pour un scénario s est donc

$$\sum_{i \in T} (C_i(\sigma_s^*, s) - C_i(\sigma, s)) = \sum_{i \in T} (\pi_s^*(i) - \pi(i))\, p_i^s$$

- On cherche donc la séquence (σ, π) qui minimise le plus grand regret :

$$\min_{(\sigma,\pi) \in \Sigma} \max_{s \in S} \sum_{i \in T} (\pi_s^*(i) - \pi(i))\, p_i^s$$

Ordonnancement robuste avec intervalles de durées - problème minimax regret absolu

Scénarios dominants

- Comment évaluer $\max_{s \in S} \sum_{i \in T} (\pi_s^*(i) - \pi(i))\, p_i^s$ alors que S est l'ensemble des points $p^s \in \mathbb{R}^{|T|}$ vérifiant $\underline{p}_i \leq p_i^s \leq \overline{p}_i$, $i \in T$?
- Scénario extrême : $p_i^s = \underline{p}_i$ ou $p_i^s = \overline{p}_i$, $\forall i \in T$

Théorème (Kouvelis et Yu (1997))

- *Le scénario qui maximise l'écart à l'optimum pour une séquence (σ, π) donnée est un scénario extrême.*
- *Pour ce scénario $\hat{s}$, $p_i^{\hat{s}} = \overline{p}_i$ si $\pi_{\hat{s}}^*(i) > \pi_i$ et $p_i^{\hat{s}} = \underline{p}_i$ si $\pi_{\hat{s}}^*(i) \leq \pi_i$*

preuve au tableau

On peut en déduire un algorithme polynomial pour trouver le scénario de plus grand regret absolu pour une séquence donnée. Mais la recherche de la séquence qui minimise le plus grand regret est NP-difficile.

Programmation linéaire en nombres entiers robuste avec recours

Variante de l'approche stochastique (voir cours 3)

Soit un ensemble de scénarios $s \in S$ avec les paramètres T_s, q_s, W_s, d_s, on peut obtenir un équivalent déterministe au problème robuste minimax (et minimax regret absolu)

$$
\begin{aligned}
& \min z \\
& z \geq c^T x + {q_s}^T y_s - z^*(s), \quad \forall s \in S \\
& Ax \geq b \\
& T_s x + W_s y_s \geq d_s, \quad \forall s \in S \\
& x_1, \ldots, x_p \in \mathbb{N} \\
& x_{p+1}, \ldots, x_{p+g} \geq 0 \\
& y_{s1}, \ldots, y_{sp'} \in \mathbb{N}, \quad \forall s \in S \\
& y_{s,p'+1}, \ldots, y_{s,p'+g'} \geq 0, \quad \forall s \in S
\end{aligned}
$$

où $Z^*(s)$ est l'optimum pour le scénario s

Optimisation Robuste

Incertitude dans les coefficients des contraintes

Formulation générale

$$
\begin{aligned}
\min_x \quad & c^T x \\
s.c. \quad & Ax \leq b
\end{aligned}
$$

Contrepartie robuste

$$
\begin{aligned}
\min_x \max_{c \in \mathcal{C}} \quad & c^T x \\
s.c. \quad & Ax \leq b \quad \forall (A, b) \in \mathcal{U}
\end{aligned}
$$

Les ensembles $\mathcal{U}$ et $\mathcal{C}$ représentent les incertitudes, c'est-à-dire TOUS les scénarios que l'on souhaite considérer. On peut facilement se ramener à une incertitude seulement sur le membre de gauche de l'inéquation.

$$\begin{array}{lll} \min\limits_{x,t} & t & \\ s.c. & c^T x - t & \leq 0 \quad \forall\, c \in \mathcal{C} \\ & Ax - bx_{n+1} & \leq 0 \quad \forall\, (A,b) \in \mathcal{U} \end{array}$$

où $t \in \mathbb{R}$ est une nouvelle variable et x_{n+1} est une variable qui doit être forcée à être égale à 1.

L'ensemble d'incertitude adopté pour modéliser les coefficients de la matrice A est un modèle par intervalles, $a_{ij} \in [\bar{a}_{ij} - \hat{a}_{ij}, \bar{a}_{ij} + \hat{a}_{ij}]$, où $\bar{a}_{ij}$ représente la valeur nominale du coefficient a_{ij} et $\hat{a}_{ij}$ (avec $\hat{a}_{ij} \geq 0$) sa déviation maximale :

$$a_{ij} = \bar{a}_{ij} + \hat{a}_{ij}\xi_{ij}$$

avec

$$\xi_{ij} \in [-1, 1]$$

Optimisation Robuste

Incertitude dans les coefficients des contraintes

Deux approches sont possibles :

- Contexte statique
- Contexte multi-étapes

Contexte statique

La décision est prise avant toute connaissance de l'incertitude.

Contexte multi-étapes

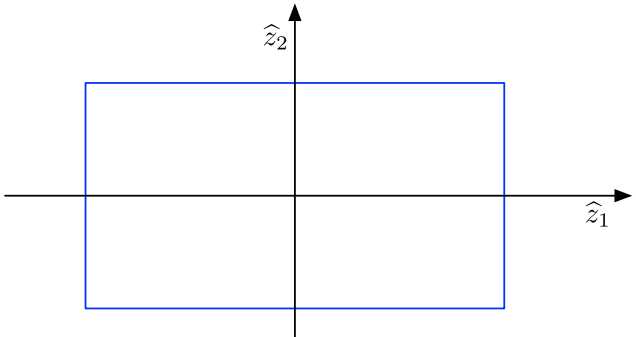
Les variables de décisions sont décomposées en plusieurs groupes, de sorte que certaines décisions peuvent être prises au fur et à mesure que les incertitudes sont révélées.

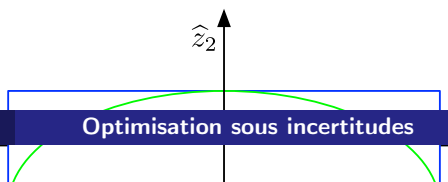
Optimisation Robuste

Incertitude dans les coefficients des contraintes : contexte statique

Dans le contexte statique, trois ensembles d'incertitude peuvent être considérés.

- incertitudes de type "box"
- incertitudes ellipsoïdales
- incertitudes polyhedrales





Incertitudes de type "box"

Le problème suivant

$$\begin{cases} \min & Cx & & \\ s.c. & Ax & \leq & b \\ & x & \geq & 0 \end{cases}$$

devient

$$\begin{cases} \min & Cx & & & & \\ s.c. & \sum_{j=1}^{n}(\bar{a}_{ij} + \hat{a}_{ij}\xi_{ij})x_j & \leq & b_i & \forall \xi_{ij} \in [-1,1] & \forall i = 1 \dots m \\ & x & \geq & 0 & & \end{cases}$$

On peut réécrire le problème précédent de la façon suivante :

$$\begin{cases} \min & Cx & & & \\ s.c. & \sum_{j=1}^{n}\bar{a}_{ij}x_j + \max_{\xi_{ij}\in[-1,1]}\{\hat{a}_{ij}\xi_{ij}x_j\} & \leq & b_i & \forall i = 1 \dots m \\ & x & \geq & 0 & \end{cases}$$

incertitudes de type "box"

Puisque $x \geq 0$

$$\begin{cases} \min & Cx & & & \\ s.c. & \sum_{j=1}^{n}(\bar{a}_{ij} + \hat{a}_{ij})x_j & \leq & b_i & \forall i = 1 \dots m \\ & x & \geq & 0 & \end{cases}$$

Remarques

- Approche plutôt naïve (Soyster, 1973)
- Elle conduit à l'optimisation sur le pire cas.
- Reste un PL
- Solution obtenue très conservative !

Example

$$\begin{cases} \max & -2x_1 + 3x_2 \\ s.c. & a_{11}x_1 + a_{12}x_2 \leq 120 \\ & a_{21}x_1 + a_{22}x_2 \geq 30 \\ & x_1 \leq 8 \\ & x_1, x_2 \geq 0 \end{cases}$$

avec $\bar{a}_{11} = -9$, $\bar{a}_{12} = 20$, $\bar{a}_{21} = 2$, $\bar{a}_{22} = 5.5$ et $\hat{a}_{11} = 6$, $\hat{a}_{12} = 5$, $\hat{a}_{21} = 1$, $\hat{a}_{22} = 0.5$.

Example

La version robuste en considérant des incertitudes de type "box" est alors :

$$\begin{cases} \max & -2x_1 + 3x_2 \\ s.c. & -3x_1 + 25x_2 \leq 120 \\ & x_1 + 5x_2 \geq 30 \\ & x_1 \leq 8 \\ & x_1, x_2 \geq 0 \end{cases}$$

La solution optimale de ce problème est $x^*_{box} = (3.75, 5.25)$.

Example

La solution optimale de ce problème est $x^*_{box} = (3.75, 5.25)$. La valeur du critère vaut alors 8.25. En comparaison $x^*_{nom} = (0, 6)$ vaut 18.

Ensemble d'incertitudes paramétrable

Il est cependant possible de contrôler le conservatisme. C'est l'idée directrice des 2 autres types d'ensemble d'incertitude. La solution n'est alors plus réalisable quelque soit le scénario mais le degré de non satisfaction d'une contrainte est paramétrable. En s'autorisant cette faible probabilité de violation des contraintes, on obtient des solutions robustes de bien meilleure qualité.

Ensemble d'incertitudes ellipsoïdal

On a toujours $a_{ij} = \bar{a}_{ij} + \hat{a}_{ij}\xi_{ij}$ et $\xi_{ij} \in [-1, +1]$ mais la déviation maximale sur une même contrainte i appartient à une ellispoïde définie par :

$$\Xi_i(\Omega_i) = \left\{ \xi_{ij} \in [-1, 1] \,|\, \sqrt{\sum_{i=1}^{n} \xi_{ij}^2} \leq \Omega_i \right\}$$

avec $\Omega_i \geq 0$ pour tout $i = 1, \ldots, m$.

Ensemble d'incertitudes ellipsoïdal

$$\begin{cases} \min & Cx & & & \\ s.c. & \sum_{j=1}^{n} \bar{a}_{ij}x_j + \max\limits_{\xi_{ij}\in\Xi_i(\Omega_i)} \{\sum_{j=1}^{n} \hat{a}_{ij}\xi_{ij}x_j\} & \leq & b_i & \forall i = 1\dots m \\ & x & & \geq & 0 & \end{cases}$$

En dualisant le problème de maximisation dans la contrainte, on obtient la reformulation suivante :

$$\begin{cases} \min & Cx & & & \\ s.c. & \sum_{j=1}^{n} \bar{a}_{ij}x_j + \Omega_i z_i & \leq & b_i & \forall i = 1\dots m \\ & \sum_{j=1}^{n} \hat{a}_{ij}^2 x_j^2 & \leq & z_i^2 & \forall i = 1\dots m \\ & z & \geq & 0 & \\ & x & \geq & 0 & \end{cases}$$

Ce problème est non linéaire mais quadratique, il existe plusieurs méthodes pour le résoudre. Si $\xi_{ij} \in [-1, +1]$ suit une loi uniforme, on peut montrer que la probabilité de violation de la contrainte ne dépasse pas $\exp(\frac{-\Omega_i^2}{2})$.

Example

Reprenons l'exemple précédent en posant $\Omega_1 = 0.8$ et $\Omega_2 = 0.8$. Le problème d'optimisation robuste s'écrit alors :

$$\begin{cases} \max & -2x_1 + 3x_2 \\ s.c. & -9x_1 + 20x_2 + 0.8z_1 \leq 120 \\ & 2x_1 + 5.5x_2 - 0.8z_2 \geq 30 \\ & 36x_1^2 + 25x_2^2 - z_1^2 \leq 0 \\ & x_1^2 + 0.25x_2^2 - z_2^2 \leq 0 \\ & x_1 \leq 8 \\ & x_1, x_2 \geq 0 \end{cases}$$

Example

On obtient la solution $x^*_{ellipse} = (1.25, 5.43)$. La valeur du critère vaut alors 13.80. La probabilité de violation reste cependant très élevée : 0.77 (principalement due au très faible nombre de variables que mettent en jeu les contraintes).

Ensemble d'incertitudes polyhedral

L'ensemble polyhedral présente des caractéristiques très proches de l'approche ellipsoïdal.

$$\Xi_i(\Gamma_i) = \left\{ \xi_{ij} \in \mathbb{R}^n | \sum_{j=1}^{n} |\xi_{ij}| \leq \Gamma_i, -1 \leq \xi_{ij} \leq 1 \right\}$$

avec $\Gamma_i \in [0, n]$ pour tout $i = 1, \ldots, m$.

Ensemble d'incertitudes polyhedral

- Si $\Gamma_i = 0$, aucune déviation n'est autorisée sur la contrainte i (on a alors la contrainte nominale).
- Si $\Gamma_i = n$, tous les paramètres sont susceptibles de dévier, on revient à une incertitude de type boîte.

Ensemble d'incertitudes polyhedral

On obtient le problème robuste suivant :

$$\begin{cases} \min & Cx & & & \\ s.c. & \sum_{j=1}^{n} \bar{a}_{ij} x_j + \max_{\xi_{ij} \in \Xi_i(\Gamma_i)} \{ \sum_{j=1}^{n} \hat{a}_{ij} \xi_{ij} x_j \} & \leq & b_i & \forall i = 1 \ldots m \\ & x & & \geq & 0 & \end{cases}$$

Résolvons le problème

$$\max_{\xi_{ij} \in \Xi_i(\Gamma_i)} \{\sum_{j=1}^{n} \hat{a}_{ij} \xi_{ij} x_j\}$$

Comme $\hat{a}_{ij} x_j \geq 0$, l'optimum est atteint pour des valeurs non négatives de ξ_{ij}. L'ensemble des scénarios possibles se simplifie en :

$$\Xi_i(\Gamma_i) = \left\{ \xi_{ij} \in \mathbb{R}^n | \sum_{j=1}^{n} \xi_{ij} \leq \Gamma_i, 0 \leq \xi_{ij} \leq 1 \right\}$$

Trouver le dual de

$$\max_{\xi_{ij} \in \Xi_i(\Gamma_i)} \{\sum_{j=1}^{n} \hat{a}_{ij} \xi_{ij} x_j\}$$

Le problème s'écrit finalement

$$\begin{cases} \min & Cx & & & \\ s.c. & \sum_{j=1}^{n} \bar{a}_{ij} x_j + \pi_i \Gamma_i + \sum_{j=1}^{n} \lambda_{ij} & \leq & b_i & \forall i = 1 \dots m, \\ & \pi_i + \lambda_{ij} & \geq & \hat{a}_{ij} x_j & \forall i = 1 \dots m, \forall = 1 \dots n \\ & \pi_i & \geq & 0 & \forall i = 1 \dots m, \\ & \lambda_{ij} & \geq & 0 & \forall i = 1 \dots m, \forall = 1 \dots n \\ & x & \geq & 0 & \end{cases}$$

Ce problème reste linéaire. Si $\xi_{ij} \in [-1, +1]$ suit une loi uniforme, on peut montrer que la probabilité de violation de la contrainte ne dépasse pas $\exp(\frac{-\Gamma_i^2}{2n})$. Pour atteindre un niveau de satisfaction des contraintes de 99%, si $n = 100$ alors Γ doit être au moins égal à 30.

Example

Reprenons l'exemple précédent en posant $\Gamma_1 = 1$ et $\Gamma_2 = 1$. Le problème d'optimisation robuste s'écrit alors :

$$\begin{cases} \max & -2x_1 + 3x_2 \\ s.c. & -9x_1 + 20x_2 + \pi_1 + \lambda_{11} + \lambda_{12} \leq 120 \\ & 2x_1 + 5.5x_2 - \pi_2 - \lambda_{21} - \lambda_{22} \geq 30 \\ & \pi_1 + \lambda_{11} - 6x_1 \leq 0 \\ & \pi_1 + \lambda_{12} - 5x_2 \leq 0 \\ & \pi_2 + \lambda_{21} - x_1 \leq 0 \\ & \pi_2 + \lambda_{22} - 0.5x_2 \leq 0 \\ & x_1 \leq 8 \\ & x_1, x_2, \pi_1, \lambda_{11}, \lambda_{12}, \pi_2, \lambda_{21}, \lambda_{22} \geq 0 \end{cases}$$

Example

On obtient la solution $x^*_{polyhedral} = (1.58, 5.37)$. La valeur du critère vaut alors 12.95. On peut retrouver le scenario qui a conduit à cette solution : $\xi_{11} = 0$, $\xi_{12} = 1$, $\xi_{21} = 0$, $\xi_{22} = -1$. La probabilité de violation reste cependant très élevée : 0.78 (principalement due au très faible nombre de variables que mettent en jeu les contraintes).

Sources

- P. Kouvelis and G. Yu. Robust Discrete Optimization and its Applications. Kluwer Academic Publishers, 1997
- M. A. Aloulou and F. Della Croce. Complexity of single machine scheduling problems under scenario-based uncertainty. Operations Research Letters 36 338-342 (2008)
- D. Bertsimas. M. Sim. The price of robustness. Operations research 52(1) p.35-53, 2004.