1

Introduction et motivations

1.1 Problématique dans le cas déterministe

On a vu dans la partie B de cet ouvrage que l'on sait associer, dans le cadre *déterministe*, les idées de l'*optimisation* avec celles issues de la *décomposition/coordination*. Partant d'un problème d'optimisation écrit dans le cadre statique [1] :

$$\min_{u \in U^{\mathrm{ad}}} \; J(u) \tag{1.1a}$$

sous la contrainte

$$\Theta(u) \in -C \; , \tag{1.1b}$$

ou dans le cadre dynamique (formulé en temps discret) :

$$\min_{(u_0,\ldots,u_{T-1},x_0,\ldots,x_T)} \; \sum_{t=0}^{T-1} L_{t+1}(x_t,u_t) + \Phi(x_T) \tag{1.2a}$$

sous les contraintes

$$\begin{cases} x_0 \;\; = x_{\mathrm{ini}} \;\; \text{connu} \; , \\ x_{t+1} = f_{t+1}(x_t,u_t) \; , \;\; t = 0,\ldots,T-1 \; , \end{cases} \tag{1.2b}$$

on suppose que la taille et/ou la complexité de ce problème sont telles que l'on ne peut pas calculer sa solution à l'aide des algorithmes d'optimisation classiques (on dit alors que l'on a affaire à un *grand système*). L'idée des techniques de décomposition et coordination consiste à remplacer le problème (1.1) ou (1.2) par une suite de problèmes dits auxiliaires, telle que la suite des solutions des ces problèmes auxiliaires converge vers la solution du problème de départ. Cette transformation se fait à l'aide d'un principe général en optimisation, appelé *Principe du Problème Auxiliaire* (**PPA**). Ce principe est de type

1. $\mathbb{U}$ et $\mathbb{V}$ sont deux espaces de Hilbert, U^{ad} est un sous-ensemble convexe fermé de $\mathbb{U}$, C est un cône convexe fermé de $\mathbb{V}$, J est une fonction définie sur $\mathbb{U}$ à valeurs réelles et Θ est une fonction définie sur $\mathbb{U}$ à valeurs dans $\mathbb{V}$.

variationnel, ce qui signifie que les algorithmes de décomposition que l'on obtient à l'aide du PPA sont dans leur essence de même nature que l'algorithme du gradient.

Appliqué à un objectif de décomposition, le PPA permet de formuler des problèmes auxiliaires dont la minimisation peut être menée de manière décomposée. Par exemple, dans le cas du problème (1.1), on peut s'arranger pour que chaque problème auxiliaire se scinde en N sous-problèmes auxiliaires ne dépendant chacun que d'un sous-vecteur de la variable u, qui se met donc sous la forme $(u_1, \ldots, u_N)$, de telle sorte que l'on a à résoudre N « petits » sous-problèmes auxiliaires plutôt qu'un seul problème auxiliaire de grande taille. Dans ce cadre déterministe, le principe de la méthode de décomposition est donc de reconstituer la solution $u^\sharp$ du problème initial par concaténation des sous-vecteurs $u_i^\sharp$, limites des suites des solutions des sous-problèmes auxiliaires. Pour un exposé détaillé des méthodes de décomposition et coordination dans le cadre déterministe, on se reportera à la partie B de cet ouvrage.

On notera que, dans le cas déterministe, il n'existe pas de différence fondamentale entre le problème statique (1.1) et le problème dynamique (1.2) : les méthodes algorithmiques mises en œuvre pour résoudre l'un ou l'autre de ces problèmes sont très semblables puisque le premier consiste à chercher une solution dans l'espace $\mathbb{U}$ alors que la solution du second se trouve dans l'espace-produit $\mathbb{U}^T$. La principale différence vient du fait que l'on peut utiliser plus ou moins habilement le temps dans la formulation et la résolution du problème (1.2), par exemple en calculant les gradients à l'aide de la technique de l'état-adjoint.

1.2 Extensions au cas stochastique

Le but de la deuxième partie de cet ouvrage est de proposer des méthodes permettant d'étendre au cas *stochastique* les techniques de décomposition et de coordination pour des problèmes de type (1.1) ou (1.2).

Dans toute la suite, les variables aléatoires seront notées avec des lettres majuscules grasses, comme par exemple $\boldsymbol{W}$, et les réalisations des variables aléatoires seront notées avec des lettres minuscules normales, comme par exemple w.

1.2.1 Cas statique

Problème.

On considère pour commencer le problème (1.1), en oubliant dans un premier temps les contraintes explicites Θ, et on suppose que la fonction J s'écrit comme l'espérance d'une fonction j dépendant d'une variable aléatoire $\boldsymbol{W}$

définie sur un espace de probabilité $(\Omega, \mathcal{A}, \mathbb{P})$ à valeurs dans un espace $\mathbb{W}$ muni d'une tribu $\mathcal{W}$:

$$J(u) = \mathbb{E}\left(j(u, \boldsymbol{W})\right) \ .$$

Le problème d'optimisation stochastique à résoudre est alors :

$$\min_{u \in U^{\mathrm{ad}}} \ \mathbb{E}\left(j(u, \boldsymbol{W})\right) \ . \tag{1.3}$$

La même valeur de u s'applique pour toutes les valeurs w que peut prendre la variable aléatoire $\boldsymbol{W}$; cette situation est qualifiée de *statique*, au sens où la valeur optimale $u^\sharp$ que l'on cherche ne s'adapte pas aux valeurs que prend $\boldsymbol{W}$ [2] : on dit que l'on est en *boucle ouverte* pour signifier que, en tant que fonction de w, la commande optimale $u^\sharp$ est une fonction *constante*.

D'un point de vue théorique, on peut calculer, en tout point u par lequel cheminera l'algorithme d'optimisation choisi pour résoudre le problème, la valeur $J(u)$ du critère et son gradient $\nabla J(u)$, si bien que le problème d'optimisation stochastique (1.3) se ramène au problème déterministe (1.1) : les méthodes de décomposition s'y appliquent alors sans difficulté particulière. Cependant, d'un point de vue pratique, chaque évaluation de $J(u)$ ou de $\nabla J(u)$ nécessite un calcul d'espérance, ce qui peut s'avérer très consommateur en temps de calcul sur un ordinateur, surtout si si l'espace $\mathbb{W}$ dans lequel la variable aléatoire $\boldsymbol{W}$ prend ses valeurs est de grande dimension...

Gradient stochastique.

Une approche mieux adaptée que la précédente à ce type de problème est celle du *gradient stochastique*, dont le principe est de faire évoluer *simultanément* le calcul de l'espérance *et* la méthode de gradient, en s'appuyant sur la méthode de Monte-Carlo pour l'évaluation des espérances. L'idée de la méthode consiste à tirer une suite $(w^{(1)}, \ldots, w^{(k)}, \ldots)$ de réalisations de la variable aléatoire $\boldsymbol{W}$ suivant sa loi de probabilité, et à faire évoluer la variable u pour chaque nouvelle valeur de l'aléa en effectuant un pas de gradient sur la fonction j en fixant $\boldsymbol{W}$ à cette valeur. L'évolution de u doit être suffisamment lente pour que le phénomène de moyenne lié à l'espérance se fasse au cours des itérations. Plus précisément, la forme générale de l'algorithme de gradient stochastique est la suivante :

- on se donne $u^{(0)} \in U^{\mathrm{ad}}$ pour initialiser l'algorithme,
- à l'itération k, on effectue un tirage $w^{(k+1)}$ de la variable aléatoire $\boldsymbol{W}$ suivant sa loi (pour le calcul de l'espérance), et on effectue un pas de gradient en u (pour l'optimisation) :

$$u^{(k+1)} = \mathrm{proj}_{U^{\mathrm{ad}}}\left(u^{(k)} - \epsilon^{(k)} \nabla_u j(u^{(k)}, w^{(k+1)})\right) \ ,$$

où $\epsilon^{(k)}$ est le terme d'une suite de réels positifs qui converge « lentement » vers zéro [3].

2. $u^\sharp$ dépend par contre de la *loi de probabilité* de la variable aléatoire $\boldsymbol{W}$.
3. car si $\epsilon^{(k)}$ converge trop vite vers zéro, l'algorithme peut s'arrêter trop tôt...

On constate que derrière le gradient stochastique se trouve une idée de nature variationnelle, que l'on devrait donc pouvoir adapter au cadre de la décomposition/coordination.

Cette seconde partie du cours va porter sur l'étude du gradient stochastique, de sa généralisation et de son efficacité. Dans la mesure où le **PPA** permet de traiter des problèmes d'optimisation avec contraintes explicites, on étendra le gradient stochastique au cas des contraintes Θ *déterministes*.

Interprétation intuitive.

On a dit que l'idée du gradient stochastique consiste à profiter des itérations d'optimisation pour effectuer le calcul d'espérance. Pour rendre cette affirmation plus concrète, on considère l'algorithme de gradient stochastique en supposant que l'ensemble U^{ad} est égal à l'espace $\mathbb{U}$ tout entier :

$$u^{(k+1)} = u^{(k)} - \epsilon^{(k)} \nabla_u j(u^{(k)}, w^{(k+1)}) \ .$$

Sommant k fois cette formule à partir d'un indice k_0 donné et supposant :
- d'une part que la fonction : $u \to \nabla_u j(u, w)$ est suffisamment régulière,
- d'autre part que les valeurs $u^{(k_0+l)}$ sont peu différentes les unes des autres pour $0 \leq l \leq k$ [4],

on obtient la première approximation suivante :

$$\begin{aligned} u^{(k_0+k)} &= u^{(k_0)} - \sum_{l=0}^{k-1} \epsilon^{(k_0+l)} \nabla_u j(u^{(k_0+l)}, w^{(k_0+l+1)}) \ , \\ &\approx u^{(k_0)} - \sum_{l=0}^{k-1} \epsilon^{(k_0+l)} \nabla_u j(u^{(k_0)}, w^{(k_0+l+1)}) \ . \end{aligned}$$

De cette dernière forme, en utilisant la proposition 1.5 (en annexe de ce chapitre), on déduit la nouvelle approximation [5] :

$$u^{(k_0+k)} \approx u^{(k_0)} - \left(\sum_{l=0}^{k-1} \epsilon^{(k_0+l)} \right) \nabla J(u^{(k_0)}) \ .$$

Sous cette forme, l'algorithme du gradient stochastique s'interprète comme une méthode de gradient classique, et il apparaît clairement que le gradient stochastique utilise les itérations d'optimisation pour reconstituer l'espérance du *gradient du critère* et non l'espérance du critère lui-même.

Remarque 1.1. Les deux approximations effectuées nécessitent des hypothèses en apparence contradictoires sur l'indice k. Cette contradiction est levée par le fait que les pas $\epsilon^{(k_0+l)}$ sont « petits » pour k_0 suffisamment « grand ».

4. ce qui veut dire que l'indice k_0 est suffisamment grand **et** que l'indice k n'est pas trop grand...

5. et il faut alors supposer que l'indice k est suffisamment grand !

1.2.2 Cas dynamique

Les problèmes d'optimisation stochastique faisant intervenir des systèmes évoluant dans le temps ne seront pas traités dans le cadre de cet ouvrage. On présente brièvement quelques caractéristiques de ces problèmes stochastiques dynamiques afin de souligner les différences avec le cas statique.

Problème.

L'extension du problème (1.2) au cas stochastique est de considérer un système dynamique soumis à chaque pas de temps à une perturbation aléatoire $\boldsymbol{W}_t$ dont les réalisations sont notées w_t. La dynamique décrivant l'évolution du système consiste en une relation définissant la condition initiale (aléatoire) du système :

$$x_0 = f_{-1}(w_0) \ , \tag{1.4a}$$

et une relation définissant l'évolution du système sur le pas de temps $[t, t+1[$:

$$x_{t+1} = f_t(x_t, u_t, w_{t+1}) \ , \quad t = 0, \dots, T-1 \ , \tag{1.4b}$$

dans laquelle le choix des indices temporels des arguments de la fonction f_t est là pour indiquer que l'on choisit au pas de temps $[t, t+1[$ la valeur de la commande u_t *avant* de connaître la valeur w_{t+1} de la perturbation $\boldsymbol{W}_{t+1}$ affectant ce même pas de temps. On dira alors que la structure d'information du problème est en *Décision-Hasard* (**DH**), car on décide d'abord de la commande, le hasard n'intervenant que dans un deuxième temps. On suppose qu'il n'y a pas de dynamique caché dans l'évolution des aléas au cours du temps, ce qui revient à dire que les bruits $\boldsymbol{W}_t$ et $\boldsymbol{W}_{t'}$ à deux pas de temps différents sont des variables aléatoires indépendantes. Le but de l'optimisation est de minimiser une fonction coût de la forme :

$$\mathbb{E}\left(\sum_{t=0}^{T-1} L_t(x_t, u_t, \boldsymbol{W}_{t+1}) + K(x_T)\right) \ , \tag{1.4c}$$

où le coût instantané L_t dépend de l'état x_t, de la commande u_t et est lui aussi perturbé par l'aléa $\boldsymbol{W}_{t+1}$, tandis que la fonction K représente une pénalisation de l'état qu'atteint le système à la fin de la période d'optimisation.

Structure d'information.

Pour donner un sens au problème, il faut préciser la nature des variables par rapport auxquelles on optimise. Dans le cas considéré ci-dessus, il parait souhaitable que la valeur de la commande u_t à mettre en œuvre sur le pas de temps $[t, t+1[$ dépende des aléas qui ont affectés le système jusqu'au début de ce pas de temps, car la connaissance de ce qui s'est passé sur le système permet en général de mieux l'optimiser. Le calcul de la commande u_t doit donc se baser sur l'ensemble des informations disponibles sur le système au

début du pas de temps $[t, t+1[$. Le fait que l'on n'utilise que les informations passées pour calculer u_t correspond à une commande respectant le principe de *causalité*, selon lequel il ne serait pas rationnel de faire dépendre la commande d'informations non disponibles au moment de la décision, comme par exemple les valeurs *futures* des perturbations.

On formalise ces considérations de la manière suivante. On suppose qu'une *observation* z_t est effectuée sur le système au début de chaque pas de temps $[t, t+1[$, cette observation s'écrivant par exemple sous la forme :

$$z_t = h_t(x_t, w_t) \ , \quad t = 0, \dots, T-1 \ . \tag{1.4d}$$

La totalité des *informations disponibles* au début du pas de temps $[t, t+1]$ est elle-même une fonction des observations passées, que l'on note :

$$y_t = C_t(z_0, \dots, z_t) \ , \quad t = 0, \dots, T-1 \ . \tag{1.4e}$$

Un cas particulier important est celui où il n'y a pas de perte d'information au cours du temps :

$$y_t = (z_0, \dots, z_t) \ .$$

On parle alors de système à *mémoire parfaite*. D'autres cas sont envisageables, comme celui de la *mémoire instantanée* où l'on ne se rappelle que de la dernière observation effectuée :

$$y_t = z_t \ .$$

Comme on l'a déjà noté, il est raisonnable de chercher la commande u_t dans la classe des fonctions ne dépendant que de l'information y_t disponible sur le système au moment où l'on prend la décision. Cette contrainte peut se mettre sous la forme fonctionnelle suivante :

$$u_t = g_t(y_t) \ . \tag{1.4f}$$

Une commande de la forme (1.4f) est dite en *boucle fermée* sur les observations, par opposition au cas de la boucle ouverte rencontré dans le cas statique où la commande est choisie indépendamment des valeurs prises par les perturbations et donc de toute information. Bien sûr, lorsque l'on n'observe rien (c'est-à-dire lorsque l'observation est la même quels que soient les aléas affectant le système), la commande u_t est associée à une fonction constante, et l'on retrouve le cas de la boucle ouverte.

Une fois fixées les fonctions de commande $g_0, \dots, g_{T-1}$, les variables x_t, z_t, y_t et u_t du problème (1.4) deviennent toutes des *variables aléatoires*. L'évaluation de l'espérance du coût (1.4c) est alors possible, et l'optimisation cherche à minimiser cette espérance par rapport aux fonctions $g_0, \dots, g_{T-1}$.

Remarque 1.2. On voit ainsi apparaître une différence fondamentale entre les situations déterministe et stochastique, puisque l'optimisation dans le premier cas se fait par rapport à des constantes, alors qu'elle se fait par rapport à des fonctions dépendant des valeurs w_t des perturbations $\boldsymbol{W}_t$ dans le second cas.

Les notions de causalité et de boucle fermée n'ont d'ailleurs aucun sens en déterministe, puisque l'évolution future du système à un pas de temps donné ne dépend que des commandes appliquées et peut donc être anticipée par l'optimisation.

Résolution.

On distingue classiquement les trois cas suivants dans l'optimisation des systèmes dynamiques stochastiques.

- *Cas markovien.* On observe sans perturbation l'état physique x_t du système [6] :

$$z_t = x_t \ , \quad \text{et} \ \ y_t \ \ll \text{compris entre} \gg \ z_t \ \ \text{et} \ \ (z_0, \dots, z_t) \ .$$

  On montre alors que les fonctions $g_0^\sharp, \dots, g_{T-1}^\sharp$ réalisant la solution optimale du problème (1.4) ne dépendent en fait que de l'état instantané du système : la loi de commande optimale sur le pas de temps $[t, t+1[$ est de la forme :

$$u_t = g_t^\sharp(x_t) \ .$$

  Pour calculer le coût optimal et les lois de commande associées, on dispose de la méthode de la *programmation dynamique*, qui consiste à résoudre, en tout point de l'espace dans lequel vit l'état physique x_t du système, une équation récurrente en temps comportant un opérateur de minimisation.
- *Cas de la structure d'information classique.* L'observation z_t que l'on fait sur le système à chaque pas de temps est de la forme générale (1.4d), mais on suppose que l'accumulation des informations au cours du temps se fait sans perte de mémoire (mémoire parfaite) :

$$z_t = h_t(x_t, w_t) \ , \quad \text{et} \ \ y_t = (z_0, \dots, z_t) \ .$$

  La résolution du problème de commande optimale est possible et se fait de la manière suivante :
  - on écrit d'abord l'équation du filtre régissant l'évolution au cours du temps de la loi de probabilité de l'état x_t conditionné par l'observation disponible y_t ;
  - on résout ensuite une équation de programmation dynamique, qui se développe dans l'espace (a priori de dimension infinie) des lois de probabilité conditionnelle de l'état.
- *Cas général.* On ne sait pas résoudre les conditions d'optimalité associées au problème.

6. On rappelle que l'on suppose que les bruits à deux pas de temps différents sont des variables aléatoires indépendantes, de telle sorte que les seules équations régissant l'évolution du système sont les équations (1.4b).

Pour un exposé détaillé de la commande optimale stochastique dans le cas markovien et dans le cas de l'information classique (aussi appelé cas de l'information incomplète), on se référera à Puterman (2009), ou encore au cours Quadrat et Viot (2000). Pour l'analyse d'un (contre-) exemple dans le cas général, on pourra consulter l'article Witsenhausen (1968).

Programmation dynamique et décomposition.

La mise en œuvre de la méthode de la programmation dynamique n'est en pratique possible que pour les systèmes dont l'état physique x_t est à valeurs dans un espace de petite dimension (inférieure ou égale à 3 pour fixer les idées). Ce phénomène est connu sous le nom de *malédiction de la dimension*. Il correspond à une difficulté d'ordre méthodologique (et non technologique) et empêche en pratique l'utilisation directe de la programmation dynamique sur la plupart des systèmes industriels et financiers en vraie grandeur.

Si l'on cherche alors, pour l'optimisation des grands systèmes dynamiques stochastiques, à marier la technique de programmation dynamique aux méthodes de décomposition/coordination, on se trouve confronté à une nouvelle difficulté, elle aussi d'ordre méthodologique. Pour l'illustrer, on considère un système se décomposant en N sous-systèmes interagissant entre eux, l'état x et la commande u du système initial se mettant respectivement sous la forme $(x_1, \ldots, x_N)$ et $(u_1, \ldots, u_N)$. On suppose que l'on est dans le cas markovien : la loi de commande du sous-système i au pas de temps $[t, t+1[$ est donc de la forme :

$$u_{i,t} = g_t(x_{1,t}, \ldots, x_{N,t}) \ ,$$

et dépend donc de l'état de *tous* les sous-systèmes. La résolution de l'équation de programmation dynamique associée au sous-système i doit être développée sur l'état du système tout entier, ce qui ne correspond pas à l'idée de formuler des sous-systèmes indépendants les uns des autres dans le processus de décomposition/coordination. Dans ce cas, c'est la classe dans laquelle on cherche les lois de commande qui interdit la décomposition du système.

Cette impossibilité s'explique par le fait que l'on cherche à décomposer l'espace de départ des fonctions g_t définissant les commandes (espace dans lequel vit l'état x) alors que la décomposition fonctionne bien lorsque l'on décompose l'espace d'arrivée (espace dans lequel vit la commande u).

Remarque 1.3. On pourrait décider arbitrairement de limiter l'optimisation à la classe des lois de commande décentralisées, de la forme :

$$u_{i,t} = g_t(x_{i,t}) \ .$$

La décomposition et la mise en œuvre de la programmation dynamique par sous-système redeviennent possibles dans certains cas, au prix d'un calcul global d'espérance (voir par exemple Delebecque et Quadrat (1978)). Mais on se contente alors de chercher une solution *sous-optimale* du problème, et il est facile de se rendre compte sur des exemples simples que la classe des

lois de commande décentralisée peut conduire à des solutions suffisamment sous-optimales pour être inacceptables. Ainsi, dans le cas de deux unités de production indépendantes ayant à satisfaire conjointement une consommation, si l'on suppose que la première unité produit à faible coût tout en étant sujette à des pannes fréquentes alors que la seconde unité produit à coût élevé sans aucune panne, il est clair que la commande optimale de la deuxième unité consiste à se substituer à la première unité en cas de panne de cette dernière, ce qui montre bien qu'ignorer l'état de la première unité dans la commande de la seconde ne peut pas être satisfaisant.

Programmation stochastique.

Puisque l'on est dans l'incapacité d'optimiser les systèmes dynamiques stochastiques de grande taille par la programmation dynamique, on se tourne vers d'autres techniques pour résoudre le problème, en particulier le formalisme de la *programmation stochastique*[7]. L'idée générale est de *ne pas transporter les lois de probabilité* dans les espaces où vivent l'état et la commande (comme le fait la programmation dynamique), mais de représenter ces variables d'état et de commande comme des variables aléatoires (vivant donc sur l'espace de probabilité original). La discrétisation du problème est alors possible (par exemple par une approche de type Monte-Carlo), mais il faut s'assurer qu'elle respecte la structure d'information associée au problème.

1.3 Un exemple statique et dynamique

On étudie un exemple mélangeant les caractéristiques des problèmes (1.3) et (1.4), et représentant un problème typique de l'optimisation stochastique dans lequel on cherche à réaliser le meilleur compromis entre des coûts d'investissement et de fonctionnement, par exemple dans un réseau de distribution de services (télécommunication, électricité), dans le transport aérien...

1. **Investissement**. On doit choisir la capacité u d'un outil de production. Ce choix est fait une fois pour toutes, en connaissant la distribution de probabilité des aléas pouvant affecter le système, mais sans connaître la valeur de l'aléa qui se réalisera : la commande u est donc en boucle ouverte, et on note $\alpha(u)$ le coût associé à l'installation de la capacité u.
2. **Fonctionnement**. Une fois l'investissement u réalisé, le fonctionnement du système est perturbé par une variable aléatoire $\boldsymbol{W}$ que l'on observe. On dispose alors d'une commande v permettant d'agir sur le réseau, à investissement u et à valeur w de $\boldsymbol{W}$ connus, afin de satisfaire le service rendu par le système. La commande v dépend de u et w, et est donc en boucle fermée sur l'aléa. Sa mise en œuvre induit un coût noté $\beta(u, v, w)$.

7. On pourra consulter le site `http://stoprog.org/` qui contient un grand nombre d'informations (présentations, livres, articles, logiciels, exemples...) relatives à la communauté de la programmation stochastique.

Le problème consiste à minimiser par rapport à u et v l'espérance du coût total, soit :

$$\min_{(u,v)} \mathbb{E}\left(\alpha(u) + \beta\left(u, v, \boldsymbol{W}\right)\right) \ . \tag{1.5}$$

On va comparer deux approches pour la résolution de ce problème, l'une intuitive (mais qui conduira à un échec), et l'autre plus raisonnée (qui fournira la solution du problème).

(A) L'application naïve du gradient stochastique à ce problème consiste à mettre en œuvre un algorithme de remise à jour simultanée des variables u et v, de la forme suivante :

- on se donne des valeurs initiales $u^{(0)}$ et $v^{(0)}$,
- au pas k de l'algorithme,
  - on tire une valeur $w^{(k+1)}$ de $\boldsymbol{W}$,
  - on effectue une étape de gradient de pas $\epsilon^{(k)}$ sur v :

$$v^{(k+1)} = v^{(k)} - \epsilon^{(k)} \nabla_v \beta(u^{(k)}, v^{(k)}, w^{(k+1)}) \ ,$$

  - on effectue une étape de gradient de pas $\rho^{(k)}$ sur u :

$$u^{(k+1)} = u^{(k)} - \rho^{(k)} \left(\nabla \alpha(u^{(k)}) + \nabla_u \beta(u^{(k)}, v^{(k+1)}, w^{(k+1)})\right) \ .$$

Cependant, cet algorithme ne fournit pas la solution en u du problème (1.5), car on peut montrer que la limite de la suite $\left\{u^{(k)}\right\}$ qu'il engendre dépend des paramètres $\epsilon^{(k)}$ que l'on a choisit pour faire évoluer l'algorithme. Dans cette approche, on a ignoré le fait que la commande v doit être en boucle fermée sur l'aléa $\boldsymbol{W}$, d'où l'échec...

(B) On suppose maintenant que l'on est capable de résoudre le problème d'optimisation lié au seul fonctionnement : pour des valeurs données $u^{(k)}$ et $w^{(k+1)}$ de l'investissement et de la variable aléatoire $\boldsymbol{W}$, on effectue la minimisation :

$$\min_{v} \beta\left(u^{(k)}, v, w^{(k+1)}\right) \ , \tag{1.6}$$

dont la solution optimale, notée $v^{(k+1)}$, est en fait une fonction $v^\sharp(u^{(k)}, w^{(k+1)})$ dépendant à la fois des valeurs $u^{(k)}$ et $w^{(k+1)}$ de l'investissement et de la perturbation. Une fois le problème du fonctionnement optimal à investissement donné résolu, on peut appliquer l'algorithme du gradient stochastique à la seule variable d'investissement u, qui est quant à elle en boucle ouverte sur l'aléa.

On est donc conduit à mettre en œuvre l'algorithme suivant :

- on se donne une valeur initiale $u^{(0)}$,
- au pas k de l'algorithme,
  - on tire une valeur $w^{(k+1)}$ de $\boldsymbol{W}$, indépendant des tirages précédents,

– on résout complètement [8] le problème de minimisation en v :

$$v^{(k+1)} = \arg\min_{v} \beta\left(u^{(k)}, v, w^{(k+1)}\right) ,$$

– on effectue une étape de gradient de pas $\epsilon^{(k)}$ sur u :

$$u^{(k+1)} = u^{(k)} - \epsilon^{(k)}\left(\nabla\alpha(u^{(k)}) + \nabla_u\beta(u^{(k)}, v^{(k+1)}, w^{(k+1)})\right) .$$

Cet algorithme fournit la solution $u^\sharp$ du problème (1.5). Une fois l'investissement optimal $u^\sharp$ déterminé, la résolution du problème de fonctionnement seul pour cet investissement et pour une valeur donnée w de la variable aléatoire $\boldsymbol{W}$ permet de calculer $v^\sharp(u^\sharp, w)$, meilleure façon de piloter le système.

Remarque 1.4. On a utilisé un résultat « classique » en optimisation qui dit que, si une fonction ϕ est le résultat de la minimisation d'une fonction J par rapport à l'un de ses arguments :

$$\phi(u) = \min_{v} J(u, v) ,$$

alors, en notant $\widehat{v}(u)$ une solution de ce problème, et sous des hypothèses « raisonnables » de convexité, de continuité et de différentiabilité de la fonction J, la fonction ϕ est elle aussi différentiable, et que son gradient est égal au gradient partiel de J par rapport à u évalué au point $\widehat{v}(u)$ qui réalise le minimum :

$$\nabla\phi(u) = \nabla_u J(u, \widehat{v}(u)) .$$

1.4 Annexe : extension du théorème de Césaro

On a utilisé, dans l'interprétation intuitive de l'algorithme du gradient stochastique, le résultat suivant qui est une généralisation du théorème de la moyenne de Césaro.

Proposition 1.5.
Soit $\left\{x^{(k)}\right\}_{k\in\mathbb{N}}$ *une suite à valeurs dans un espace de Hilbert* $\mathbb{X}$*, qui converge* ***en moyenne*** *vers* μ *:*

$$\lim_{N\to+\infty}\left(\frac{1}{N}\sum_{k=1}^{N} x^{(k)}\right) = \mu .$$

Soit $\left\{\rho^{(k)}\right\}_{k\in\mathbb{N}}$ *une suite de réels positifs, décroissante et qui converge vers* 0*. On suppose que la suite réelle positive* $\left\{\epsilon^{(k)}\right\}_{k\in\mathbb{N}}$ *définie par :*

8. en supposant que ce problème admet une unique solution

$$\epsilon^{(k)} = k\big(\rho^{(k)} - \rho^{(k+1)}\big) \ ,$$

est telle que $\epsilon^{(k)}$ *soit le terme général d'une série* ***divergente****. Alors, on a :*

$$\lim_{N\to\infty} \left(\frac{\displaystyle\sum_{k=1}^{N} \rho^{(k)} x^{(k)}}{\displaystyle\sum_{k=1}^{N} \rho^{(k)}} \right) = \mu \ .$$

Preuve. On se ramène pour commencer au cas $\mu = 0$ par changement de variable $x^{(k)} \rightsquigarrow x^{(k)} - \mu$. On pose :

$$y^{(k)} = \frac{1}{k} \sum_{l=1}^{k} x^{(l)} \ ,$$

qui par hypothèse converge vers 0. De la définition du pas $\epsilon^{(k)}$ et de la convergence de $\rho^{(k)}$ vers 0, on déduit que :

$$\rho^{(k)} = \sum_{l=k}^{+\infty} \frac{\epsilon^{(l)}}{l} \ .$$

Soit alors un entier $N \in \mathbb{N}$. Pour tout $k \leq N$, on peut écrire :

$$\rho^{(k)} = \rho_N^{(k)} + \rho^{(N+1)} \qquad \text{avec} \qquad \rho_N^{(k)} = \sum_{l=k}^{N} \frac{\epsilon^{(l)}}{l} \ . \tag{1.7}$$

Avec ces notations, on a :

$$\sum_{k=1}^{N} \epsilon^{(k)} y^{(k)} = \sum_{k=1}^{N} \frac{\epsilon^{(k)}}{k} \sum_{l=1}^{k} x^{(l)} = \sum_{k=1}^{N} x^{(k)} \sum_{l=k}^{N} \frac{\epsilon^{(l)}}{l} = \sum_{k=1}^{N} \rho_N^{(k)} x^{(k)} \ , \tag{1.8}$$

et donc (en appliquant l'égalité précédente avec $x^{(k)} = 1$ pour tout k) :

$$\sum_{k=1}^{N} \epsilon^{(k)} = \sum_{k=1}^{N} \rho_N^{(k)} \ , \tag{1.9}$$

Utilisant (1.7), (1.8) et (1.9), on a :

$$\begin{aligned}
\frac{\left\|\sum_{k=1}^{N}\rho^{(k)}x^{(k)}\right\|}{\sum_{k=1}^{N}\rho^{(k)}} &= \frac{\left\|\sum_{k=1}^{N}\rho_N^{(k)}x^{(k)}+\rho^{(N+1)}\sum_{k=1}^{N}x^{(k)}\right\|}{\sum_{k=1}^{N}\rho_N^{(k)}+N\rho^{(N+1)}} \\
&\leq \frac{\left\|\sum_{k=1}^{N}\rho_N^{(k)}x^{(k)}\right\|}{\sum_{k=1}^{N}\rho_N^{(k)}+N\rho^{(N+1)}} + \frac{\left\|\rho^{(N+1)}\sum_{k=1}^{N}x^{(k)}\right\|}{\sum_{k=1}^{N}\rho_N^{(k)}+N\rho^{(N+1)}} \\
&\leq \frac{\left\|\sum_{k=1}^{N}\rho_N^{(k)}x^{(k)}\right\|}{\sum_{k=1}^{N}\rho_N^{(k)}} + \frac{\left\|\rho^{(N+1)}\sum_{k=1}^{N}x^{(k)}\right\|}{N\rho^{(N+1)}} \\
&\leq \frac{\left\|\sum_{k=1}^{N}\epsilon^{(k)}y^{(k)}\right\|}{\sum_{k=1}^{N}\epsilon^{(k)}} + \frac{1}{N}\left\|\sum_{k=1}^{N}x^{(k)}\right\| .
\end{aligned}$$

Dans le membre de droite de cette dernière inégalité, le premier terme tend vers 0 en vertu du théorème « classique » de la moyenne de Césaro (car $\epsilon^{(k)}$ est le terme d'une série divergente), et le second terme tend lui aussi vers 0 par hypothèse. On en déduit le résultat annoncé.

Remarque 1.6. Les hypothèses faites dans la propriété précédente sont *consistantes*. Typiquement, le pas $\rho^{(k)}$ est de la forme $\frac{1}{k^\gamma}$, avec $\gamma > 0$ (pour assurer la convergence de la suite vers zéro). Un calcul au premier ordre montre que le pas $\epsilon^{(k)}$ a pour expression :

$$\epsilon^{(k)} = k\frac{1}{k^\gamma}\left(\frac{\gamma}{k} + o(k^{-1})\right) ,$$

et est donc aussi de la forme $\frac{\gamma}{k^\gamma}$. Les suites $\left\{\rho^{(k)}\right\}_{k\in\mathbb{N}}$ et $\left\{\epsilon^{(k)}\right\}_{k\in\mathbb{N}}$ sont donc de même nature. Elles respectent les hypothèses de la propriété 1.5 (et en particulier celle sur la divergence de la série de terme général $\epsilon^{(k)}$) dès que l'on a : $0 < \gamma \leq 1$.