Université Paris Saclay

Ecole nationale supérieure de Techniques avancées

EDF R&D, Paris Saclay

Apport de l'approche stochastique pour l'optimisation d'une batterie sur le marché spot et le marché des services système

---

Présenté par :
Nidhal GAMMOUDI

Encadré par :
Clemence ALASSEUR
Nadia OUDJANE
Xavier WARIN

Avril – Septembre 2017

Table des matières

Remerciements

J'adresse mes plus sincères remerciements à mes encadrants de stage, Nadia OUDJANE, Clemence ALASSEUR et Xavier WARIN pour leurs aide. Malgré leurs responsabilités, ils étaient toujours disponibles pour me diriger et me donner de leurs savoir-faire. C'est en grande partie grâce à eux que je garderais un très bon souvenir de mon expérience à OSIRIS.

Plus généralement, je tiens à remercier tout le personnel des groupes R33 et R36 d'OSIRIS qui m'ont accueilli et offert un environnement de travail convivial.

Je remercie également mon encadrant pédagogique Monsieur Pierre CARPENTIER et Monsieur Filippo SANTAMBROGIO, responsbale du master *OPIMIZATION* de Paris-Saclay que j'ai suivi cette année, pour avoir accepté d'évaluer mon travail.

Je remercie enfin l'ensemble du corps professoral de l'ENSTA pour la qualité de la formation que j'ai reçue, ainsi que les responsables du master *OPIMIZATION* qui m'ont beaucoup aidé pour mes choix de parcours.

Chapitre 1

Introduction

"Developments in the electricity industry over the past few years have increased interest in energy storage."[1, p. 1]

Parmi les raisons qui ont contribué à ce développement, on cite essentiellement :

— Introduction des marchés définissant les coûts et les profits des services de stockage.
— Augmentation de la contribution des énergies renouvelables dans le système énergétique, d'où la nécessité de développer des techniques permettant le stockage de cette énergie pour en profiter.
— Amélioration des caractéristiques et des performances des batteries (physiques, coûts ...).

Dans ce stage, on étudie le cas d'une batterie de stockage d'électricité : on cherche à prendre des décisions de charge et de décharge pour gérer son usage.
On doit tenir compte de deux aspects : technique et économique.

Nous allons nous intéresser plus particulièrement à la planification d'usage de cette batterie, dans le but de maximiser les gains sur les différents marché d'électricité. Ce ci nécessite des techniques économiques (choisir un modèle mathématique qui modélise l'évolution aléatoire des prix pendant la période d'étude) et mathématiques (optimisation : trouver les stratégies optimales pour maximiser le gain).

Parmi les différents services que cette batterie peut assurer, on considère dans notre problème : arbitrage sur le marché d'énergie (spot) et les services systèmes (ancillary services).

— Arbitrage sur le marché spot d'énergie : il s'agit de charger la batterie quand les prix d'électricité sont moins élévés et de la décharger quand les prix augmentent pour maximiser un gain économique.
— Services systèmes : il s'agit de mobiliser des réserves de puissance qui seront utilisées dans les cas de variations de la demande et de la production pour contribuer à maintenir l'équilibre offre/demande.

Les services systèmes, appelés aussi services de régulation consistent à être flexible et

à agir en temps réel pour minimiser, soit même annuler, les différences entre la demande et l'offre. On distingue ainsi, deux types de régulation :

— Régulation à la hausse (Regulation-up) : consiste à fournir de l'énergie pour satisfaire une demande qui a dépassé l'offre.
— Régulation à la baisse (Regulation-down) : consiste à stocker de l'énergie dans la batterie dans le cas d'un excès (l'offre dépasse la demande d'où l'absorption d'une quantité limitée de l'énergie en excès).

Les incertitudes pour ce type de problèmes sont importantes : aléas de consommation, de production et des prix sur les différents marchés d'énergie (marché spot et marché des services systèmes). Pour cette raison, on adopte une approche stochastique au lieu des méthodes déterministes souvent utilisées pour résoudre ce type de problème.

Le contrôle stochastique a de nombreuses applications dans des problèmes de gestion de l'énergie, d'économie et de finance. C'est une situation où on fait face à des systèmes dynamiques évoluant dans des conditions d'incertitude et il s'agit donc de prendre des décisions à chaque date afin d'optimiser un critère économique.

Dans ce contexte et en tenant compte de l'évolution aléatoire des prix qui se fait d'un jour à l'autre, on s'intéresse notamment à la maximisation du gain résultant de l'usage de la batterie sur les deux marchés, spot et services systèmes, pendant une durée de J jours, $j = 1..J$.

Dans ce rapport, on présentera dans l'ordre :

— Le modèle qu'on a établi pour ce problème.
— La résolution mathématique du problème (Programmation dynamique, régression $\cdots$)
— L'implémentation des méthodes numériques pour la résolution (librairie StOpt).
— Quelques tests effectués.

Mots clés

Electricité, stockage, marché spot (d'électricité), marché de régulation, contrôle stochastique, programmation dynamique, régression.

Chapitre 2

Modélisation du problème

2.1 Paramètres

On distingue deux types de paramètres : physiques et économiques.

2.1.1 Paramètres physiques

Les paramètres physiques représentent les caractéristiques de la batterie étudiées :
— P_{max} : puissance maximale de charge et de décharge de la batterie.
— E_{min} : niveau de stock minimal d'énergie dans la batterie.
— E_{max} : niveau de stock maximal d'énergie dans la batterie.
— ρ_c : coefficient d'efficacité de charge de la batterie.
— ρ_d : coefficient d'efficacité de décharge de la batterie.

La puissance maximale de charge/décharge, P_{max}, reflète généralement les limites de puissance de l'onduleur [1] dans la batterie ou les limites thermiques du stockage électrochimique moyen dans la batterie (les batteries produisent de la chaleur lorsqu'elles sont chargées et déchargées).

On tient compte dans notre modèle d'un niveau de stock minimal d'énergie à ne pas dépasser. En effet, pour certains types de batterie (comme les batteries à Lithuim) le cycle de vie subit une dégradation remarquable si le niveau de stock atteint des valeurs très basses.

Les coefficients d'efficacité ρ_c et ρ_d représentent les pertes d'énergie possibles lors de la charge et la décharge. Le produit $\rho_c\rho_d$ est le roundtrip efficiency de la batterie, c'est à dire le pourcentage d'énergie produite par la batterie après un cycle charge-décharge.

Remarque : Dans le cas où la charge/la décharge du stock d'électricité est idéal (se

1. Un onduleur est un dispositif d'électronique de puissance permettant de générer des tensions et des courants alternatifs à partir d'une source d'énergie électrique de tension ou de fréquence différente. C'est la fonction inverse d'un redresseur

fait sans pertes), il suffit de mettre le coefficient d'efficacité correspondant égal à 1. Ces coefficient sont sans unité.

2.1.2 Paramètres économiques

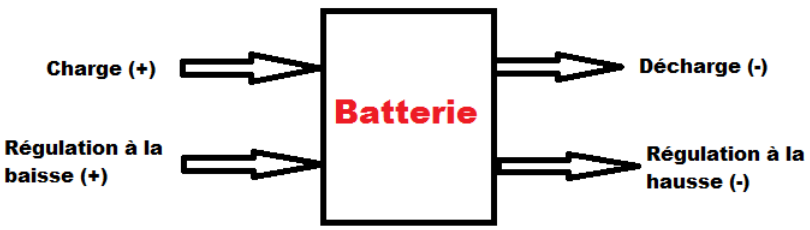
— PR : facteur de pénalité de non-satisfaction de la régulation (à la hausse ou bien à la baisse).
— PS : facteur de pénalité de non-satisfaction des engagements avec le marché spot pour les quantités à charger ou bien décharger sur le marché spot.

Remarque : PR et PS sont aussi sans unité. Ces deux paramètres vont apparaître dans la fonction objectif du problème d'optimisation. Pour exprimer un coût économique, ils vont être multipliés par le prix d'énergie. Il s'agit donc de pourcentage par rapport au prix sur le marché d'énergie.
Ces deux paramètres expriment également la priorité entre les engagements spot et régulation. En effet, pour privilégier un engagement par rapport à l'autre, il suffit d'augmenter le facteur de pénalité correspondant.

2.2 Variables de décision

Hypothèse : Pour les décisions de type charge/décharge, on travaille en bilan. C'est à dire, pour chaque heure de la journée, on a le droit soit de charger soit de décharger. Grâce à cette hypothèse, on considère une seule variable algébrique pour ce type de décisions, ce qui permet de réduire le nombre de variables et par suite la taille du problème.

Convention de signe :
On adopte le signe + pour l'énergie qui entre dans la batterie (ayant pour source les décisions de charge ou de regulation-up) et le signe - pour l'énergie sortante (venant de la décision de décharge ou celle de regulation-down).



Pour gérer l'usage de la batterie suivant les stratégies d'optimisation, on se sert des variables de décision suivantes :

— $e_{j,t}$: quantité d'énergie échangée entre la batterie et le marché spot pendant l'heure t du jour j [KWh] (de signe positif si achetée, négatif sinon).

— $r^u_{j,t}$: capacité de puissance de régulation à la hausse que le gestionnaire s'engage à fournir durant l'heure t du jour j [KW-h] (l'indice u veut dire Regulation-up).

— $r^d_{j,t}$: capacité de puissance de régulation-down que le gestionnaire s'engage à absorber durant l'heure t du jour j [KW-h] (l'indice d veut dire Regulation-down).

On note par $u_{j,t}$ le contrôle à l'heure t du jour j, $u_{j,t} = (e_{j,t}, r^u_{j,t}, r^d_{j,t})$, pour $j = 1..J$ et $t = 0..T-1$, où $T = 24$.

2.2.1 Offre sur le marché spot : Charge/Décharge

La variable de décision $e_{j,t}$ peut être décomposée en deux autres variables auxiliaires, $e^+_{j,t}$ et $e^-_{j,t}$:

— $e^+_{j,t}$: C'est la quantité d'énergie achetée par la batterie pendant l'heure t du jour j [KWh].

— $e^-_{j,t}$: C'est la quantité d'énergie vendue par la batterie sur le marché spot pendant l'heure t du jour j [KWh].

Ces deux variables sont déduites à partir de $e_{j,t}$ par les équations non-linéaires suivantes :

$$e^+_{j,t} = max\{0, e_{j,t}\} \tag{2.1}$$
$$e^-_{j,t} = -min\{0, e_{j,t}\} \tag{2.2}$$

Ainsi, $e_{j,t}$ se met sous la forme suivante :

$$e_{j,t} = e^+_{j,t} - e^-_{j,t} \tag{2.3}$$

2.2.2 Offre sur le marché de régulation

Prendre une décision de régulation revient à allouer deux puissances sur une heure :

— Régulation à la baisse : On s'engage à absorber en cas d'excés. En effet, pour une heure t, on aloue une puissance r^d_t (l'unité est le kW-h). Une telle décision veut dire que la batterie doit être capable d'absorber, sur l'heure t, toute quantité d'énergie comprise entre les valeurs 0 et r^d_t. Une pénalité sera calculée à l'heure t si la batterie n'arrive pas à satisfaire cet engagement.

— Régulation à la hausse : On s'engage à fournir de l'énergie en cas de déficit. En effet, pour une heure t, on aloue une puissance r^u_t (l'unité est le kW-h). Une telle

décision veut dire que la batterie doit être capable de fournir, sur l'heure t, toute quantité d'énergie comprise entre les valeurs 0 et r_t^u. La non-satisfaction de cet engagement engendre une pénalité à cette heure t.

Remarque : kW-h, qui est une unité pour les services système, est différente de kWh qui mesure l'énergie. En effet, kW-h veut dire une puissance allouée sur une heure.

2.3 Variables d'état

Notre modèle a besoin également des variables d'état suivantes : une variable pour le stock d'énergie dans la batterie, deux variables pour les prix d'énergie sur le marché spot et marché de régulation et une dernière variable qui modélise le besoin effectif du réseau d'énergie en services de régulation.

- $x_{j,t}$: quantité d'énergie totale stockée dans la batterie au début de l'heure t du jour j [kWh].
- $p^s_{j,t}$: prix de l'énergie sur le marché spot à l'heure t du jour j [euros/kWh].
- $p^r_{j,t}$: prix de la capacité de puissance de régulation up et down sur le marché des services système à l'heure t du jour j [euros/kW-h] (On suppose que les deux types de régulation up et down se font au même prix).
- $\delta_{j,t}$ [2] : ratio d'écart caractérisant le besoin effectif du réseau pour la régulation up ou down pendant l'heure t du jour j, pour $t = 0..T-1$ et $j = 0..J$. Ce ratio varie entre -1 et 1.

L'état du système, à l'heure t du jour j, est par suite représenté par la variable $\{y_{j,t}\}_{t=0}^{T-1}$, tel que $y_{j,t} = (x_{j,t}, \delta_{j,t}, p^s_{j,t}, p^r_{j,t})$.

2.3.1 Prix d'énergie et de capacité de puissance de régulation

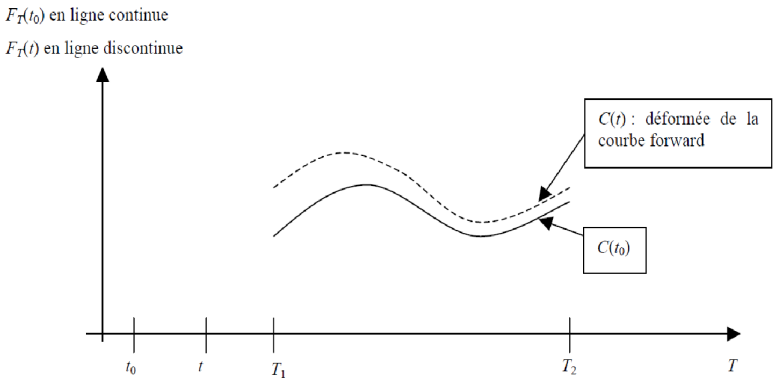
Pour les prix d'énergie et de capacité de puissance de régulation, on adopte un modèle multiplicatif, en exprimant le prix $F(t,T)$ comme le produit entre le prix initial et un terme aléatoire positif dit de diffusion :

$$F(t,T) \quad = \quad F(t_0,T) \times diffusion(t,T) \tag{2.4}$$

Où :

- t_0 est la date de début de diffusion
- t est la date de cotation.
- T est la date du début de livraison.

2. On reviendra sur cette notation dans la partie Ratio d'écart



Plus précisément, il s'agit d'un modèle à un seul facteur exponentiel, dont l'équation différentielle est donnée par :

$$dF(t,T) = F(t,T)\sigma \exp(-a(T-t))dw_t \tag{2.5}$$

Avec :
— a : le coefficient de retour à la moyenne non nul.
— σ : la volatilité du modèle.
— w_t : un brownien.
On pose $X(t,T) = \log F(t,T)$ et on applique la formule d'Itô à $X(t,T)$:

$$dX(t,T) = \frac{1}{F(t,T)}dF(t,T) - \frac{1}{2F(t,T)^2}\langle dF(t,T), dF(t,T)\rangle$$

Où,

$$\langle dF(t,T), dF(t,T)\rangle = F(t,T)^2\sigma^2 \exp(-2a(T-t))dt$$

Donc :

$$dX(t,T) = \sigma \exp(-a(T-t))dw_t - \frac{\sigma^2}{2}\exp(-2a(T-t))dt$$

Par suite, $X(t,T)$ s'exprime comme suit :

$$X(t,T) = X(0,T) + \exp(-a(T-t))Y_t - \sigma^2 \int_0^t \exp(-2a(T-t))ds$$

Les prix futures sont donc exprimés comme suit :

$$F(t,T) = F(0,T)\exp[\exp(-a(T-t))Y_t - \frac{\sigma^2}{2a}\exp(-2a(T-t))[1-\exp(-2at)]]$$

Y_t est l'état markovien et il est donné par :

$$Y_t = \int_0^t \sigma \exp(-a(t-s))dw_s \tag{2.6}$$

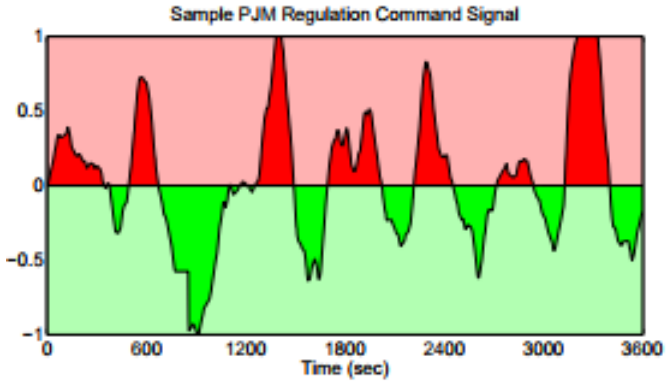
Notion de martingale
Le processus aléatoire $F(t,T)$ vérifie pour tout t :

$$E\left[F(t,T)\right] \ = F(0,T) \tag{2.7}$$

On dit donc que le processus $F(.,T)$ est une martingale.

2.3.2 Ratio d'écart

En réalité, le signal de régulation est continu sur une heure et il varie entre -1 et 1. En effet, dans le graphe ci-dessous, les valeurs 1 et -1 représentent respectivement la totalité de la puissance reservée pour la régulation à la baisse et à la hausse. Donc, c'est le signal de régulation qui oscille entre ces deux valeurs limites.



Dans notre cas, on a discuté essentiellement 3 idées pour modéliser cette variable :
— partager l'heure en des sous intervalles (minute, seconde...) : ceci aurait pour conséquence d'augmenter énormèment la complexité du problème.
— considérer deux variables indépendantes entre 0 et 1 (une pour la régulation à la hausse et l'autre pour la régulation à la baisse) : cette idée donne une fausse information parce qu'elle ne respecte pas la notion d'ordre dans le temps. On peut donc être pénalisé alors qu'on a bien satisfait nos engagements et inversement.
— considérer pour chaque heure une seule variable entre -1 et 1, qui représente le bilan entre la régulation à la hausse et la régulation à la baisse $\delta_{j,t}$: ce qu'on va retenir pour la suite.

De même, vu qu'on va considérer la variable d'état $\delta_{j,t}$ comme le bilan entre l'énergie entrante et l'énergie sortante, elle peut être décomposée en introduisant deux autres variables auxiliaires $\delta^+_{j,t}$ et $\delta^-_{j,t}$:

— $\delta^-_{j,t}$: le ratio d'écart caractérisant le besoin réel du réseau en cas de régulation up (déficit énergétique).
— $\delta^+_{j,t}$: le ratio d'écart caractérisant le besoin réel du réseau en cas de régulation down (excès d'énergie).

Ces deux variables sont déduites à partir de $\delta_{j,t}$ par les équations non-linéaires suivantes :

$$\delta^+_{j,t} = max\{0, \delta_{j,t}\} \tag{2.8}$$
$$\delta^-_{j,t} = -min\{0, \delta_{j,t}\} \tag{2.9}$$

Ainsi, $\delta_{j,t}$ se met sous la forme suivante :

$$\delta_{j,t} = \delta^+_{j,t} - \delta^-_{j,t} \tag{2.10}$$

Remarque : Pour mieux comprendre la notion du ratio d'écart $\delta_{j,t}$:

D'abord, en suivant la même convention de signe pour la batterie, le signe de $\delta_{j,t}$ indique le type de régulation qu'on a satisfait :
— $\delta_{j,t} > 0$: c'est le cas de la régulation à la baisse : on charge.
— $\delta_{j,t} < 0$: il s'agit de la régulation à la hausse : on décharge.

De plus,
— $\delta^-_{j,t} r^u_{j,t}$: c'est la quantité d'énergie que la batterie doit fournir pour satisfaire la régulation à l'heure t du jour j.
— $\delta^+_{j,t} r^d_{j,t}$: c'est la quantité d'énergie que la batterie doit absorber pour satisfaire la régulation à l'heure t du jour j.

Conclusion : La quantité d'énergie réellement appelée par le service opérateur pour assurer la régulation (à la hausse ou à la baisse), à l'heure t du jour j, est donnée par :

$$\Delta E_{j,t} = \delta^+_{j,t} r^d_{j,t} - \delta^-_{j,t} r^u_{j,t} \tag{2.11}$$

2.4 Variables de pénalisation

Hypothèse : On suppose que les engagements sur le marché de régulation sont prioritaires par rapport à ceux sur le marché d'énergie.

Pour qu'on puisse déterminer la dynamique des variables d'état, on doit tout d'abord introduire deux types de variables auxiliaires, en faisant l'hypothèse que la satisfaction des besoins du marché de régulation (up et down) est prioritaire par rapport à la satisfaction de ceux du marché spot (charge et décharge). Il s'agit des variables qui expriment la différence entre les décisions de charge/décharge et de régulation prises à l'avance et le pouvoir de la batterie en temps réel (c'est à dire la batterie doit être capable de contenir

la quantité d'énergie issue de la décision de charge ou celle de la régulation à la baisse, ou bien fournir la quantité d'énergie à décharger ou nécessaire pour satisfaire la régulation à la hausse).

2.4.1 Non-satisfaction de la régulation

Les deux variables auxiliaires qu'on présente dans cette sous partie modélisent les quantités d'énergie que la batterie n'arrive pas à fournir ou à absorber pour accomplir ses missions de régulation :

- $n^u_{j,t}$: représente la différence entre la quantité d'énergie nécessaire pour la régulation-up et la quantité maximale d'énergie que la batterie est capable de fournir à l'heure t du jour j. Si cette différence est négative (batterie capable de satisfaire la régulation-up), on renvoie zéro.
- $n^d_{j,t}$: représente la différence entre la quantité d'énergie nécessaire pour la régulation-down et la quantité d'énergie que la batterie est capable d'absorber à l'heure t du jour j. Si cette différence est négative (batterie capable de satisfaire la régulation-down), on renvoie également zéro.

Ces deux grandeurs sont directement déterminées à partir du ratio de besoin effectif $\delta_{j,t}$ et la quantité d'énergie totale stockée dans la batterie au début de l'heure t du jour j, par les deux équations non-linéaires suivantes :

$$n^u_{j,t} = max\{0, -\delta_{j,t} r^u_{j,t} - \rho_d(x_{j,t} - E_{min})\} \tag{2.12}$$

$$n^d_{j,t} = max\{0, \delta_{j,t} r^d_{j,t} - (E_{max} - x_{j,t})/\rho_c\} \tag{2.13}$$

2.4.2 Non-satisfaction des engagements sur le marché spot

Une fois les quantités de régulation non satisfaites déterminées, on peut les utiliser pour exprimer les différences entre les décisions d'énergie à échanger entre la batterie et le marché spot et ses capacités de charge et décharge après la régulation.

Il s'agit des pénalités pour la non-satisfaction des engagements sur le marché spot :

— $m^D_{j,t}$: représente la différence entre la quantité décidée pour être déchargée sur le marché spot et la quantité maximale que la batterie peut décharger après la régulation à l'heure t du jour j. Si cette différence est négative (batterie capable de décharger la quantité décidée), on renvoie zéro.

— $m^C_{j,t}$: représente la différence entre la quantité décidée pour être chargée auprès du marché spot et la quantité maximale d'énergie que la batterie peut recevoir après la régulation à l'heure t du jour j. Si cette différence est négative (batterie capable de recevoir la quantité décidée), on renvoie zéro.

$$m^D_{j,t} = \mathbb{1}_{\{e_{j,t}<0\}} max\{0, -e_{j,t} - \rho_d(x_{j,t} - E_{min}) \\ - \rho_c(\delta^+_{j,t} r^d_{j,t} - n^d_{j,t}) + 1/\rho_d(\delta^-_{j,t} r^u_{j,t} - n^u_{j,t})\} \quad (2.14)$$

$$m^C_{j,t} = \mathbb{1}_{\{e_{j,t}>0\}} max\{0, e_{j,t} - (E_{max} - x_{j,t})/\rho_c \\ + \rho_c(\delta^+_{j,t} r^d_{j,t} - n^d_{j,t}) - 1/\rho_d(\delta^-_{j,t} r^u_{j,t} - n^u_{j,t})\} \quad (2.15)$$

Remarque : $\forall t = 0..T-1$, $\delta_{j,t}$ n'est connu que vers la fin de l'heure t.

2.5 Dynamique de l'état de charge

Finalement, on peut écrire la dynamique de la SOC (state of charge) de la batterie :

$$x_{j,t+1} = f(x_{j,t}, \delta_{j,t}, u_{j,t}) = x_{j,t} + \rho_c(e^+_{j,t} - m^C_{j,t} + \delta^+_{j,t} r^d_{j,t} - n^d_{j,t}) \\ - 1/\rho_d(e^-_{j,t} - m^D_{j,t} + \delta^-_{j,t} r^u_{j,t} - n^u_{j,t}) \quad (2.16)$$

Avec :

— $x_{j,t}$: l'ancien état de charge.

— $\rho_c(e^+_{j,t} - m^C_{j,t} + \delta^+_{j,t} r^d_{j,t} - n^d_{j,t})$: la quantité d'énergie qui sera chargée dans la batterie après intéraction avec le marché spot et celui de régulation.

— $1/\rho_d(e^-_{j,t} - m^D_{j,t} + \delta^-_{j,t} r^u_{j,t} - n^u_{j,t})$: la quantité d'énergie de la batterie qui sera déchargée après inéraction avec le marché spot et celui de régulation.

2.6 Contraintes

En prenant les décisions sur le marché spot et celui de service système, on doit respecter la puissance maximale de charge et de décharge de la batterie P_{max}.

On exprime les contraintes en fonction de $e^+_{j,t}$ et $e^-_{j,t}$ grâce aux équations (1) et (2), avec Δt=1 h :

$$0 \leq \frac{e^+_{j,t}}{\Delta t} + r^d_{j,t} \leq P_{max} \tag{2.17}$$

$$0 \leq \frac{e^-_{j,t}}{\Delta t} + r^u_{j,t} \leq P_{max} \tag{2.18}$$

L'énergie totale présente est limitée par les bornes de stockage de la batterie comme suit :

$$E_{min} \leq x_{j,t} \leq E_{max} \tag{2.19}$$

Les variables de décision concernant la régulation sont positives ou nulles :

$$r^u_{j,t}, r^d_{j,t} \geq 0 \tag{2.20}$$

Le ratio d'écart $\delta_{j,t}$ est borné entre -1 et 1 :

$$\delta_{j,t} \in [-1, 1] \tag{2.21}$$

Remarque : La contrainte (2.19) est facultative. En effet, grâce à la dynamique f qui fait intervenir les variables auxiliaires, l'état de charge restera toujours entre les bornes E_{min} et E_{max}.

2.7 Fonction objective

Le gain de la batterie de l'heure t du jour j, pour $t = 0..T-1$ et $j = 1..J$, est donné par la formule suivante :

$$\begin{aligned} g_{j,t}(x_{j,t}, u_{j,t}, \delta_{j,t}, p^s_{j,t}, p^r_{j,t}) = -p^s_{j,t} e_{j,t} + p^r_{j,t}(r^u_{j,t} + r^d_{j,t}) - PR \times p^s_{j,t}(n^u_{j,t} + n^d_{j,t}) \\ - PS \times p^s_{j,t}(m^C_{j,t} + m^D_{j,t}) \end{aligned} \tag{2.22}$$

En effet :
- Le terme $-p^s_{j,t} e_{j,t}$ représente le revenu des échanges avec le marché spot.
- Le terme $p^r_{j,t}(r^u_{j,t} + r^d_{j,t})$ représente le revenu de l'engagement pour la régulation up et down.
- Le terme $PR \times p^s_{j,t}(n^u_{j,t} + n^d_{j,t})$ représente la pénalité en cas de non satisfaction de la régulation up ou down.
- Le terme $PS \times p^s_{j,t}(m^C_{j,t} + m^D_{j,t})$ représente la pénalité en cas de non satisfaction des engagements avec le marché spot.

Chapitre 3

Formulation et résolution mathématique du problème d'optimisation

3.1 Problème global

3.1.1 Notations

Pour le problème global, sur J jours, on considère un aléa **markovien** des prix $w_j = (p_j^s, p_j^r)$ et l'aléa des ratios d'écart δ_j, avec :
— δ_j représente les ratios d'écart du jour j, $\delta_j = \{\delta_{j,t}\}_{t=1}^{T}$, pour $j = 1..J$.
— p_j^s et p_j^r représente les prix spot et régulation pour un jour j, $p_j^s = \{(p_{j,t}^s\}_{t=0}^{T-1}$ et $p_j^r = \{(p_{j,t}^r\}_{t=0}^{T-1}$.

Pour le contrôle, il s'agit maintenant d'une matrice de contrôle u_j sur le jour j, de taille $T \times 3$: $u_j = \{u_{j,t}\}_{t=0}^{T-1}$.

On note également par x_j le stock initial de la batterie du jour j : $x_j = x_{j,0}$.

Ainsi, on introduit une nouvelle fonction h gérant la dynamique de l'état de charge du niveau x_j au niveau x_{j+1}. En fait, h est le résultat de l'application de la dynamique f (transition entre les heures du même jour) T-fois, et à chaque fois, f prend en paramètre l'ancien état de charge, le ratio d'écart et le contrôle sur cette heure. Autrement, l'écriture :

$$\begin{aligned} x_{j,1} &= f(x_j, \delta_{j,1}, u_{j,0}) \\ x_{j,2} &= f(x_{j,1}, \delta_{j,2}, u_{j,1}) \\ &\vdots \\ x_{j+1} &= f(x_{j,T-1}, \delta_{j,T}, u_{j,T-1}) \end{aligned}$$

est équivalente à l'écriture :

$$x_{j+1} \quad = \quad h(x_j, \delta_j, u_j) \tag{3.1}$$

3.1.2 Formulation du problème

On note par G_j la somme de tous les gains du jour j :

$$G_j(x_j, u_j, \delta_j, w_j) \quad = \quad \sum_{t=0}^{T-1} g_{j,t}(x_{j,t}, u_{j,t}, \delta_{j,t}, p^s_{j,t}, p^r_{j,t}) \tag{3.2}$$

Le problème d'optimisation global sur les J jours se met sous la forme suivante :

$$\underset{u_1..u_J}{\text{maximize}} \quad \mathbb{E}\left[\sum_{j=1}^{J} G_j(x_j, u_j, \delta_j, w_j) + V_{J+1}(x_{J+1})|x_1, w_1\right] \tag{3.3}$$

$$\text{s.c} : \begin{cases} x_{j+1} = h(x_j, \delta_j, u_j) \\ E_{min} \leq x_j \leq E_{max}, \\ \text{u}_j \in \sigma(w_1, ..., w_j, x_j) \end{cases}$$

V_{J+1} est appelé la "valeur de fin de jeu".

Remarque

La contrainte $u_j \in \sigma(w_1, ..., w_j, x_j)$ exprime la mesurabilité : c'est l'information dont on dispose au moment de prendre les décisions. En prenant les décisions du jour j, on connaît les prix d'énergie et de capacité de puissance de régulation pour toute la journée j ainsi que les journées passées $1...j-1$: $p^s_{k,t}$ et $p^r_{k,t}$ sont connus pour tout $k = 1...j$ et $t = 0...T-1$. On connaît également tous les ratios d'écart pour les jours passés $1...j-1$: $\delta_{k,t}$ sont connus pour tout $k = 1...j$ et $t = 0...T-1$.

Par contre, ce sont les ratios d'écart du jour j, $\delta_{j,t}$ pour tout $t = 0...T-1$, qui sont inconnus au moment de prendre les décisions.

Autre formulation

On définit, pour un contrôle u et une réalisation $w = (p^s, p^r)$, $\tilde{G}_j(x, u, w)$ par :

$$\tilde{G}_j(x, u, w) \quad = \quad \mathbb{E}_{\delta_j}\left[G_j(x, u, w, \delta_{j+1})\right] \tag{3.4}$$

Pour tout $j = 1..J$, on a l'égalité suivante :

$$\begin{aligned} \mathbb{E}\left[G_j(x_j, u_j, w_j, \delta_j)|x_1, w_1\right] &= \mathbb{E}\left[\mathbb{E}_{\delta_j}\left[G_j(x_j, u_j, w_j, \delta_{j+1})|w_j = w\right]|x_1, w_1\right] \\ &= \mathbb{E}\left[\mathbb{E}_{\delta_j}\left[G_j(x_j, u_j, w_j, \delta_j)\right]|x_1, w_1\right] \\ \mathbb{E}\left[G_j(x_j, u_j, w_j, \delta_j)|x_1, w_1\right] &= \mathbb{E}\left[\tilde{G}_j(x_j, u_j, w_j)|x_1, w_1\right] \end{aligned}$$

Grâce à cette transformation, le problème (3.3) se ramène au problème suivant :

$$\underset{u_1..u_J}{\text{maximize}} \quad \mathbb{E}\left[\sum_{j=1}^{J} \tilde{G}_j(x_j, u_j, w_j) + V_{J+1}(x_{J+1})|x_1, w_1\right] \tag{3.5}$$

$$\text{s.c} : \begin{cases} x_{j+1} = h(x_j, \delta_j, u_j) \\ E_{min} \leq x_j \leq E_{max}, \\ \mathrm{u}_j \in \sigma(w_1, ..., w_j, x_j) \end{cases}$$

3.1.3 Programmation dynamique

Pour un stock initial x et un scénario des prix $w = (p^s, p^r)$, la valeur de Bellman du problème global, $\mathbb{V}_j(x, w)$, se définit comme suit,

$$\mathbb{V}_j(x, w) \quad = \quad \max_{u_j..u_J} \mathbb{E}\left[\sum_{i=j}^{J} \tilde{G}_i(x_i, u_i, w_i) + V_{J+1}(x_{J+1})|x_j = x, w_j = w\right] \tag{3.6}$$

D'une façon générale, $\forall\ j = 1..J-1$, on a :

$$\begin{aligned}\mathbb{V}_j(x, w) &= \max_{u_j..u_J} \mathbb{E}\left[\sum_{i=j}^{J} \tilde{G}_i(x_i, u_i, w_i) + V_{J+1}(x_{J+1})|x_j = x, w_j = w\right] \\ &= \max_{u_j}\left\{\tilde{G}_j(u_j, w) + \max_{u_{j+1}..u_J} \mathbb{E}\left[\sum_{i=j+1}^{J} \tilde{G}_i(x_i, u_i, w_i) + V_{J+1}(x_{J+1})|x_j = x, w_j = w\right]\right\}\end{aligned}$$

En utilisant la loi des probabilités totales, on se ramène à l'égalité suivante :

$$\begin{aligned}\mathbb{V}_j(x, w) = \max_{u_j}\Big\{&\tilde{G}_j(u_j, w) + \\ &\mathbb{E}\left[\max_{u_{j+1}..u_J} \mathbb{E}\left[\sum_{i=j+1}^{J} \tilde{G}_i(x_i, u_i, w_i) + V_{J+1}(x_{J+1})|x_{j+1}, w_{j+1}\right]|x_j = x, w_j = w\right]\Big\}\end{aligned}$$

Finalement,

$$\mathbb{V}_j(x, w) \quad = \max_{u_j}\Big\{\tilde{G}_j(x_j, u_j, w) + \mathbb{E}\left[\mathbb{V}_{j+1}(x_{j+1}, w_{j+1})|x_j = x, w_j = w\right]\Big\}$$

Ou bien :

$$\mathbb{V}_j(x, w) \quad = \max_{u_j}\Big\{\tilde{G}_j(x_j, u_j, w) + \mathbb{E}\left[\mathbb{V}_{j+1}(h(x, \delta_{j+1}, u_j), w_{j+1})|w_j = w\right]\Big\}$$

3.2 A l'intérieur du jour : Optimisation sur les heures

3.3 Formulation du problème

En programmation dynamique, pour faire un pas vers l'arrière (pas backward) afin de passer du jour $j+1$ au jour j, il faut résoudre un problème d'optimisation sur les heures du jour j, en prenant comme valeur finale le gain à partir du jour $j+1$. Ce problème d'optimisation peut être exprimé sous la forme suivante :

$$\underset{u_{j,0}..u_{j,T-1}}{\text{maximize}} \quad \mathbb{E}\left[\sum_{t=0}^{T-1} g_{j,t}(x_{j,t}, u_{j,t}, \delta_{j,t}, p^s_{j,t}, p^r_{j,t}) + \mathbb{V}_{j+1}(x_{j+1}, w_{j+1})|x_j, w_j\right] \tag{3.7}$$

$$\text{s.c} : \begin{cases} x_{j,t+1} = f(x_{j,t}, u_{j,t}, \delta_{j,t}) \\ E_{min} \leq x_{j,t} \leq E_{max} \\ \mathrm{u}_{j,t} \in \sigma(x_{j,t}, p^s_{j,t}, p^r_{j,t}) \end{cases}$$

Remarque : La contrainte $u_{j,t} \in \sigma(x_{j,t}, p^s_{j,t}, p^r_{j,t})$ veut dire que pour prendre la décision de l'heure t, $u_{j,t}$, on doit connaître les prix d'énergie à cette heure, $p^s_{j,t}$ et $p^r_{j,t}$, ainsi que le stock initial $x_{j,t}$. Connaître le stock $x_{j,t}$ revient à connaître l'ancien stock $x_{j,t-1}$, le contrôle et le ratio d'écart de l'heure précédente $u_{j,t-1}$ et $\delta_{j,t-1}$.

On s'intéresse donc à maximiser le gain total du jour j. Du coup, le problème d'optimisation initial est décomposé en des sous problèmes d'optimisation sur les heures.

3.4 Programmation dynamique

D'une façon générale, pour $t = 0..T-1$, la fonction valeur de l'heure t du jour j pour un stock $x_t = x$ et un scénario des prix w, $\mathbb{H}_{j,t}(x, w)$, est donnée par :

$$\mathbb{H}_{j,t}(x,w) = \max_{u_{j,t}..u_{j,T-1}} \mathbb{E}\left[\sum_{\tau=t}^{T-1} g_{j,\tau}(x_{j,\tau}, u_{j,\tau}, \delta_{j,\tau}, p^s_{j,\tau}, p^r_{j,\tau}) + \mathbb{V}_{j+1}(x_{j+1}, w_{j+1})|x_{j,t} = x, w_j = w\right]$$

Par le principe de la programmation dynamique, $\mathbb{H}_{j,t}(x, w)$ peut s'exprimer d'une autre manière :

$$\mathbb{H}_{j,t}(x,w) = \max_{u_{j,t}}\Big\{\mathbb{E}\left[g_{j,t}(x_{j,t}, u_{j,t}, \delta_{j,t}, p^s_{j,t}, p^r_{j,t})\right] + \max_{u_{j,t+1}..u_{j,T-1}} \mathbb{E}\left[\sum_{\tau=t+1}^{T-1} g_{j,\tau}(x_{j,\tau}, u_{j,\tau}, \delta_{j,\tau}, p^s_{j,\tau}, p^r_{j,\tau}) + \mathbb{V}_{j+1}(x_{j+1}, w_{j+1})|x_{j,t} = x, w_j = w\right]\Big\}$$

En utilisant la loi des probabilités totales, on se ramène à l'égalité suivante :

$$\mathbb{H}_{j,t}(x,w) = \max_{u_{j,t}} \Big\{ \mathbb{E}\left[g_{j,t}(x_{j,t}, u_{j,t}, \delta_{j,t}, p^s_{j,t}, p^r_{j,t}) \right] + \mathbb{E}\left[\max_{u_{t+1}..u_{T-1}} \mathbb{E}\left[\sum_{\tau=t+1}^{T-1} g_{j,\tau}(u_{j,\tau}, \delta_{j,\tau}) + \mathbb{V}_{j+1}(x_{j+1}, w_{j+1}) | x_{j,t+1}, w_j = w \right] | x_{j,t} = x, w_j = w \right] \Big\}$$

Finalement, on arrive à écrire l'équation de programmation dynamique :

$$\mathbb{H}_{j,t}(x,w) = \max_{u_{j,t}} \Big\{ \mathbb{E}\left[g_{j,t}(x_{j,t}, u_{j,t}, \delta_{j,t}, p^s_{j,t}, p^r_{j,t}) \right] + \mathbb{E}\left[\mathbb{H}_{j,t+1}(x_{j,t+1}, w_j) | x_{j,t} = x, w_j = w \right] \Big\}$$

En remplaçant $x_{j,t+1}$ par son expression , on obtient :

$$\mathbb{H}_{j,t}(x,w) = \max_{u_{j,t}} \Big\{ \mathbb{E}\left[g_{j,t}(x_{j,t}, u_{j,t}, \delta_{j,t}, p^s_{j,t}, p^r_{j,t}) \right] + \mathbb{E}\left[\mathbb{H}_{j,t+1}(f(x_{j,t}, u_{j,t}, \delta_{j,t+1}), w_j) | x_{j,t} = x, w_j = w \right] \Big\}$$

Ou encore,

$$\mathbb{H}_{j,t}(x,w) = \max_{u_{j,t}} \Big\{ \mathbb{E}\left[g_{j,t}(x_{j,t}, u_{j,t}, \delta_{j,t}, p^s_{j,t}, p^r_{j,t}) | \right] + \mathbb{E}\left[\mathbb{H}_{j,t+1}(f(x_{j,t}, u_{j,t}, \delta_{j,t+1}), w_j) | w_j = w \right] \Big\}$$

Une fois cette boucle backward finie et $\mathbb{H}_{j,0}(x,w)$ calculée, on revient au problème global pour déduire la fonction valeur $\mathbb{V}_j(x,w)$ grâce à l'égalité suivante :

$$\mathbb{V}_j(x,w) \quad = \mathbb{H}_{j,0}(x,w) \tag{3.8}$$

Chapitre 4

Résolution et implémentation numérique

4.1 Librairie StOpt

Pour la résolution numérique, on utilise *StOpt*, une librairie C++ qui permet de résoudre des problèmes d'optimisation stochastique en finance et industrie.

StOpt fournit les méthodes de résolution suivantes :
— Des méthodes de Programmation Dynamique basées sur la méthode de Monte Carlo avec Regression.
— Des méthodes de Semi-Lagrangian pour les équations Hamilton-Jacobi-Bellman.
— Des méthodes SDDP (Stochastic Dual Dynamic Programming) pour résoudre des problèmes stochastiques de gestion de stock.

4.2 Simulation des aléas

La première tâche était d'implémenter un simulateur d'aléa : une classe C++ nommée *Simulateur.h*.

Ce simulateur nous permet de générer un nombre donné à l'avance de simulations pour nos variables d'état probabilistes selon leurs lois de probabilité (des simulations de Monte Carlo) pour les différentes dates d'optimisation.
Dans ce travail, on se sert d'autres librairies :
— Librairie *Eigen* pour les élements de type vecteur, tableau ou matrice.
— Librairie *geners* qui nous permet de sauvegarder nos simulations à chaque pas de temps et de les restaurer si besoin (cas du backward).
— Librairie *boost* pour les lois de probabilités.
En effet, pour chaque pas de temps et en allant en forward, on élabore des simulations pour les variables aléatoires et on sauvegarde ces simulations dans un

fichier de type geners. Etant ouvert en lecture et écriture, on peut grâce à ce fichier restaurer les simulations pour n'importe quel pas de temps : ceci est utile pour les méthodes de Programmation Dynamique lorsqu'on parcourt le temps à l'inverse (Backward) pour calculer les valeurs de Bellman.

4.2.1 Simulation des prix

Processus d'Ornstein-Uhlenbeckh

Pour le modèle de prix adopté, on rappelle la formule de l'état markovien Y_t :

$$Y_t = \int_0^t \sigma \exp(-a(t-s))dw_s \tag{4.1}$$

En temps discret, on a :

$$\begin{array}{ll} & Y_{t_k} = \int_0^{t_k} \sigma \exp(-a(t_k - s))dw_s \\ \\ \Rightarrow & \exp(-a\Delta t)Y_{t_k} = \int_0^{t_k} \sigma \exp(-a(t_{k+1} - s))dw_s \\ \Rightarrow & Y_{t_{k+1}} = \exp(-a\Delta t)Y_{t_k} + \int_{t_k}^{t_{k+1}} \sigma \exp(-a(t_{k+1} - s))dw_s \\ \Rightarrow & Y_{t_{k+1}} = \exp(-a\Delta t)Y_{t_k} + \Gamma\mathcal{N}(0,1) \end{array}$$

Avec $\Gamma = \sigma\sqrt{\frac{1-\exp(-2a\Delta t)}{2a}}$.

Il s'agit donc d'un processus d'ORNSTEIN-UHLENBECK : la valeur de la variable aléatoire Y au pas du temps t_{k+1}, $Y_{t_{k+1}}$, s'obtient à partir de celle au pas de temps précédent t_k (effet mémoire) multipliée par un terme d'atténuation puis en ajoutant un terme d'innovation aléatoire.

La simulation de ce processus est faite dans le constructeur de la classe *Simulateur.h* suivant le pseudo-code suivant :

Algorithm 1 Simulation du processus d'ORNSTEIN-UHLENBECK

Données: a, σ, Δt et Y_{t_k} (l'ancien état)

1: **Pour** $t = 0$ to $NbStep$ **Faire**
2: On calcule $\Gamma = \sigma\sqrt{\frac{1-\exp(-2a\Delta t)}{2a}}$.
3: $expActu = \exp(-a\Delta t)$.
4: **Pour** $i = 0$ to $NbSimul - 1$ **Faire**
5: $increment = \Gamma \times \mathcal{N}(0,1)$.
6: $Y_{t_{k+1}} = expActu \times Y_{t_k} + increment$.
7: **Fin pour**
8: Sauvegarder les simulations obtenues dans le fichier geners.
9: **Fin pour**

Prix Spot et Propriété de martingalité

Le prix spot est celui obtenu au dernier moment où se fait l'essentiel des échanges avant l'échéance de livraison d'électricité. Il s'agit du prix d'énergie pour une heure du lendemain avec une échéance glissante, car elle change à chaque jour de cotation. Grâce à cette définition, le prix spot a pour écriture :

$$S(t) = F_{t+1jour}(t) \tag{4.2}$$

Si on dispose des prix juste avant la livraison sans le délai d'un journée on aurait :

$$S(t) = F_t(t) \tag{4.3}$$

Parmi les méthodes qu'on a implémentées dans la classe Simulateur.h, la méthode *FromParticlesToSpot* permet de récupérer les simulations du prix spot. On a vérifié, pour un nombre de simulations $nbSimul$ [1] $= 1000$ que la propriété de martingalité est valide, c'est à dire :

$$\frac{1}{nbSimul} \sum_{i=1}^{nbSimul} S^i(t) \approx F(0,t) \tag{4.4}$$

4.2.2 Simulation du ratio d'écart

Les ratios d'écart δ pour n'importe quel heure sont aléatoires et ils sont indépendants entre eux. Numériquement, on a choisi la loi uniforme $\mathcal{U}[0,1]$ pour établir des simulations de cette variable d'état :

— Pour chaque heure, on génère un nombre $nbSimul$ de simulations d'une variable aléatoire suivant la loi uniforme.
— Comme cette variable simulée est entre 0 et 1, on fait une transformation pour avoir un ratio d'écart entre -1 et 1.

4.3 Optimisation

Pour la partie Optimisation, on a développé une classe C++, appelée *Optimiseur.h*. Cette classe contient essentiellement les deux méthodes suivantes :

— StepOptimize : Optimisation sur un pas d'un jour.
— StepSimulateForward : Simulation sur un pas d'un jour.

Optimiseur.h contient d'autres fonctions secondaires qu'on utilise dans les deux méthodes précédentes. Par exemple, il y a des fonctions pour le calcul du gain, l'interpolation et la dynamique.

1. nbSimul désigne le nombre de simulations

4.3.1 Fonctions secondaires

Dans cette première partie, on présente les méthodes de *Optimiseur.h* spécifiques à notre problème, et qui seront utilisées dans les méthodes principales de cette classe.

Parmi ces fonctions :
— Fonction *dynamique* : Calcul de la dynamique de l'état de charge sur les heures,f : $x_{j,t+1} = f(x_{j,t}, \delta_{j,t}, u_{j,t})$.
— *daydynamique* : Calcul de la dynamique de l'état de charge sur les jours, h, à partir du contrôle et des ratios d'écart sur toute la journée : $x_{j+1} = h(x_j, \delta_j, u_j)$.
— Fonction *gaininstantané* : Calcul de la valeur du gain horaire pour une seule valeur de δ : $g_{j,t}(x_{j,t}, u_{j,t}, \delta_{j,t}, p^s_{j,t}, p^r_{j,t})$.
— Fonction *gainmoyen* : Calcul de la moyenne empirique des gains horarires :

$$\mathbb{E}\left[g_{j,t}(x_{j,t}, u_{j,t}, \delta_{j,t}, p^s_{j,t}, p^r_{j,t})\right] = \frac{1}{nbSimul} \sum_{i=1}^{nbSimul} g_{j,t}(x_{j,t}, u_{j,t}, \delta^i_{j,t}, p^s_{j,t}, p^r_{j,t})$$

— Fonction *valinterpolate* : A partir du contrôle, de la valeur de δ et du point initial du stock, elle calcule le nouvel état de stock et retourne la valeur interpolée de la fonction valeur : $\mathbb{H}(f(x, u, \delta), w)$.
— Fonction *meaninterpolate* : A partir du contrôle, de toutes les simulations de δ et du point initial de stock et grâce à la fonction *valinterpolate*, elle calcule la moyenne (par rapport à δ) des valeurs interpolées de la fonction valeur : $\frac{1}{nbSimul} \times \sum_{i=1}^{nbSimul} \mathbb{H}(f(x, u, \delta^i), w)$.

4.3.2 Optimisation Backward

Introduction

La méthode StepOptimize permet de faire un pas d'optimisation par le principe de la programmation dynamique : Il s'agit donc d'un pas Backward. La durée de ce pas est égal à un jour.

A l'intérieur de cette méthode, on gère une boucle sur toutes les heures de la journée pour résoudre le problème d'optimisation sur les heures. Comme on l'a expliqué avant, la résolution du problème d'optimisation sur les heures se fait également par le principe de la programmation dynamique. Ainsi, la boucle sur les heures est à l'inverse : c'est à dire on commence par la dernière heure de la journée jusqu'à la première heure. C'est la fonction valeur obtenue pour la première heure qui sera la sortie de cet algorithme, autrement c'est la fonction valeur du jour j, suivant l'égalité suivante :

$$\mathbb{V}_j(x, w) \quad = \mathbb{H}_{j,0}(x, w) \tag{4.5}$$

Problème journalier et la méthode de régression

A l'intérieur du jour, il y a deux types de problème d'optimisation à résoudre :
— Problème de la dernière heure du jour $t = 23h$: la valeur finale est une espérance conditionnelle.
— Pour $t = 0..22h$, il s'agit d'un problème d'optimisation avec des espérances normales

En effet, pour $t = T-1 = 23h$, l'équation de programmation dynamique à résoudre est :

$$\mathbb{H}_{j,T-1}(x,w) = \max_{u_{j,T-1}} \Big\{ \mathbb{E}\left[g_{j,T-1}(x_{j,T-1}, u_{j,T-1}, \delta_{j,T-1}, p^s_{j,T-1}, p^r_{j,T-1}) \right] + \mathbb{E}\left[\mathbb{V}_{j+1}(f(x, u_{j,T-1}, \delta_{j,T-1}), w_{j+1}) | w_j = w \right] \Big\}$$

On rappelle l'hypothèse qui dit que les ratios d'écart sont deux à deux indépendants. Donc, δ_j n'interviendra pas dans le calcul de l'espérance conditionnelle $\mathbb{E}\left[\mathbb{V}_{j+1}(f(x, u_{j,T-1}, \delta_{j,T-1}), w_{j+1}) | w_j = w\right]$.
Cette espérance est calculée par la méthode de régression par rapport aux simulations des prix de la dernière heure du jour j.
Le calcul de l'espérance conditionnelle va servir à l'initialisation de la matrice VB, dont le nombre de lignes est égal à celui des points de la grille des états de charge, et le nombre de colonnes est égal au nombre de simulations :

$$VB(x,i) \quad = \mathbb{E}\left[\mathbb{V}_{j+1}(x, w_{j+1}) | \{p^s_{j,T-1}, p^r_{j,T-1}\}^i \right]$$

VB désigne la Valeur de Bellman.

Pour se faire, on se sert de deux classes C++ de la librairie StOpt qui sont *LocalLinearRegression.h* et *ContinuationValue.h*.

Algorithm 2 Calcul de l'espérance conditionnelle

Données: X (grille des états de charge), $Optimalflow_{j+1}$ (valeurs de Bellman au pas de temps suivant pour tout point de X, et pour toute simulation des prix du jour $j+1$), Simulateur (objet de notre classe *Simulateur.h*)

1: On crée un opérateur de régression (objet de la classe *LocalLinearRegression.h*).
2: On récupère les simulations des prix pour la dernière heure du jour j (grâce à la méthode *getParticles* de la classe *Simulateur.h*).
3: On met à jour l'opérateur de régression par la méthode *updateSimulations* de la classe *LocalLinearRegression.h* (cette méthode prend en paramètre les simulations déjà récupérées).
4: On crée l'opérateur de la valeur continue "contvalue" par le constructeur de la classe *ContinuationValue.h*.
5: **Pour** $x \in X$ **Faire**
6: $VB(x)$=contvalue.getAllSimulations(x)
7: **Fin pour**

Remarques :

— Le constructeur de la classe *ContinuationValue.h* prend en paramètre la grille, l'opérateur de régression et les valeurs de la fonction à régresser c'est à dire $\mathbb{V}_{j+1}(x, w_{j+1})$ qui sont contenues dans la matrice $Optimalflow_{j+1}$.

— La méthode *getAllSimulations* de la classe *ContinuationValue.h* et qui prend en paramètre le point x de la grille, retourne la valeur de l'espérance conditionnelle calculée en ce point pour toutes les simulations.

Une fois que VB est initialisée, le problème à l'intérieur du jour devient comme un problème "déterministe" : On ne fait que des moyennes empiriques vu que les ratios d'ecart sont indépendants entre eux.
Pour $t = 0..22h$, l'équation de programmation dynamique à résoudre est :

$$\mathbb{H}_{j,t}(x,w) = \max_{u_{j,t}} \Big\{ \mathbb{E}\left[g_{j,t}(x_{j,t}, u_{j,t}, \delta_{j,t}, p^s_{j,t}, p^r_{j,t}) \right] + \mathbb{E}\left[\mathbb{H}_{j,t+1}(f(x, u_{j,t}, \delta_{j,t}), w) | w_j = w \right] \Big\}$$

D'une façon plus générale, et après avoir calculer l'espérance conditionnelle à la dernière heure, un pas Backward d'optimisation, d'une durée égale à une heure, à l'intérieur de la journée est exprimée comme suit :

$$VB_{j,t}(x,(p^s_{j,t},p^r_{j,t})) = \max_{u_{j,t}}\Big\{\frac{1}{nbSimul}\sum_{i=1}^{nbSimul} g_{j,t}(x_{j,t},u_{j,t},\delta^i_{j,t},p^s_{j,t},p^r_{j,t}) + \frac{1}{nbSimul}\sum_{i=1}^{nbSimul} VB_{j,t+1}(f(x_{j,t},u_{j,t},\delta^i_{j,t}),p^s_{j,t+1},p^r_{j,t+1})\Big\}$$

Avec x est un point de la grille, $\delta^i_{j,t}$, pour $i = 1..nbSimul$, sont les simulations de $\delta_{j,t}$ à l'heure t.

Interpolation

Les valeurs optimales du flow à travers les jours et les valeurs de Bellman entre les heures ne sont calculées que pour des valeurs particulières de stock d'énergie, il s'agit des points de la grille des états de charge.
Grâce à la dynamique f, en partant d'un point de la grille, et pour les différentes valeurs de δ simulées et du contrôle, on va tomber sur des points qui n'appartiennent pas à la grille. Il faut donc interpoler ...

Dans la librairie StOpt, il y a des classes définissants des différents types d'interpolation à l'instar de l'interpolation linéaire (*LinearInterpolator.h*), spectrale (*InterpolatorSpectral.h*), de Legendre (*LegendreInterpolator.h*) ...

Dans notre cas, pour calculer la valeur interpolée, on se sert d'un interpolateur linéaire en suivant les instructions suivantes :

Algorithm 3 Calcul de la valeur interpolée

Données: X (grille des états de charge), stockpoint (un tableau contenant les coordonnées du point de stock où on interpole), data (un tableau qui contient les valeurs de la fonction qu'on interpole aux points de la grille)

1: On génère un interpolateur "Interpolar" pour la grille au point de stock considéré par appel à la méthode *createInterpolator*.
2: ValeurInterpolée=Interpolar.apply(data)

Algorithme StepOptimize

Algorithm 4 StepOptimize

Données: X (grille des états de charge), stock initial x_0, $Optimalflow_{j+1}$ (valeurs de Bellman au pas de temps suivant pour tout point de X), opérateur de Valeur Continue

On crée l'espace de commandes U
On simule N scénarios de prix et du ratio δ pour toute la journée. (On fait appel à un simulateur d'aléas de type Backward [2])
Initialisation de la matrice de VB par régression par rapport aux simulations de la dernière heure.
Initialisation de la matric $flow$, $flow = Optimalflow_{j+1}$
Pour $t = 23$ to 0 **Faire**
  Initialisation de la matrice $ArbitrageMax$ à -∞
  Pour $x \in X$ **Faire**
    Pour $u \in U$ **Faire**
      Pour $s = 1$ to $nbSimul$ **Faire**
        $Arb = (gainmoyen + meanInterp(VB))(x, s)$
        Si $Arb > ArbitrageMax(x, s)$ **Alors**
          $ArbitrageMax(x, s) = Arb$
          $Optimalflow_{j,t}(x, s)$=$gainmoyen(x, s)$+$meanInter\,[flow]\,(x, s)$
        Finsi
        $VB = ArbitrageMax$, $flow = Optimalflow_{j,t}$
      Fin pour
    Fin pour
  Fin pour
Fin pour
On sauvegarde les valeurs de Bellman pour tous les pas de temps.
On interpole $Optimalflow_{j,0}$ au point x_0.
return $Optimalflow_j$ ($\equiv Optimalflow_{j,0}$) et VB pour toutes les heures du jour.

4.3.3 Simulation

La méthode StepSimulateForward, en exploitant les valeurs de Bellman pour toutes les heures du jour, fait un pas forward d'une durée égale à un jour et permet de calculer, pour un stock initial $x_{j,0}$ et un scénario de prix w_j, la valeur optimale du gain du jour j ainsi que le contrôle optimal correspondant.

Remarque : Puisque le ratio d'écart de l'heure t, $\delta_{j,t}$, n'est connu que vers la fin de cette heure, dans la boucle sur les heures de l'algorithme StepSimulateForward, et en passant de l'heure t à l'heure $t + 1$, on simule aléatoirement une valeur entre -1 et 1 et on suppose que $\delta_{j,t}$ est égal à cette valeur.

Algorithm 5 StepSimulateForward

Données: X (grille des états de charge), stock initial x_0, scénario de prix w_j, VB (valeurs de Bellman de toutes les heures du jour en tout point de X pour ce scénario de prix)

On crée l'espace de commandes U
$totalgain = 0$
Pour $t = 0$ *to* 23 **Faire**
  $data = VB(t)$ (valeurs de Bellman à l'heure t en tous points de la grille)
  $realgain(t) = -\infty$
  On récupère les simulations de $\delta_{j,t}$.
  Pour $u \in U$ **Faire**
    $gain = gainmoyen + meanInterp(data)$
    Si $gain > realgain(t)$ **Alors**
      $optimalcontrol = u$
      $realgain(t) = gain$
    Finsi
  Fin pour
  $totalgain+ = gainmoyen(x_0, optimalcontrol)$
  On simule δ et on met à jour le stock $x_0 = dynamique(x_0, optimalcontrol, \delta)$.
Fin pour
return $totalgain$, x_0 (stock modifié), $optimalcontrol$ et δ.

Chapitre 5

Tests

Paramètres physiques et économiques de la batterie

Paramètre	Valeur
E_{min}	2 KWh
E_{max}	6 KWh
P_{max}	5 KW
ρ_c	0.8
ρ_d	0.9
PR	1.15
PS	0.85

Création de l'espace d'états de charge et des commandes

Pour les deux types de commandes, charge/décharge et puissance de régulation, on utilise des grilles régulières unidimentionnelles grâce à la classe *OneDimRegularSpaceGrid.h* de StOpt.

— Pour la charge/décharge, la valeur minimale de la grille est égal $E_{min} - E_{max}$ avec un pas égal à $2 \times \frac{E_{max}-E_{min}}{nbStep1}$, où $nbStep1$ est le nombre des pas pour cette grille.

— Pour les puissances de régulation, on utilise la même grille dont la valeur minimale est 0 et le pas est égal à $\frac{P_{max}}{nbStep2}$, avec $nbStep2$ est le nombre des pas pour cette grille.

Pour les test, on a pris $nbStep1 = nbStep2 = 5$. Ce qui veut dire 6 valeurs possibles sur chaque grille. Au total, le cardinal de l'espace de commandes U est égal à 6^3.

De même, pour la grille des états de charge, on a pris un nombre de pas égal à 5,

ce qui veut dire 6 points sur cette grille.

Test de l'optmisation-simulation

Dans ce test, on essaie de voir la différence entre la valeur optimale obtenue par optimisation (backward) et celle obtenue par simulation (forward).

Ce test est fait pour un jour, 3 jours et une semaine, en changeant à chque fois le nombre de simulations *nbSimul*. Le stock initial de la batterie est égal à 2.8 KWh.

Simulateur des aléas

Pour l'optimisation, on utilise un simulateur de type backward et un simulateur de type forward pour la simulation. En effet, il suffit de modifier la valeur d'une variable booléenne passée comme paramètre au constructeur de la classe *Simulateur.h*. Cette variable indique si on est dans un mode forward ou backward.

Pour les différents types de simulateurs d'aléa, on prend la même valeur de volatilité des prix : $\sigma^s = \sigma^r = 0.94$.

Pour la phase simulation, on calcule également l'estimateur de l'écart type à la moyenne : $S = \sqrt{\frac{1}{nbSimul-1}\sum_{i=1}^{nbSimul}(X_i - \overline{X})^2}$, où $\overline{X} = \frac{1}{nbSimul}\sum_{i=1}^{nbSimul} X_i$ est la moyenne empirique.
L'intervalle de confiance à 95 % pour la moyenne empirique est directement déduit : $\left[\overline{X} - 1.96\frac{S}{\sqrt{nbSimul}}, \overline{X} + 1.96\frac{S}{\sqrt{nbSimul}}\right]$.

Résultats

Nombre de jours	Nombre de simulations	Valeur d'optimisation	Valeur de simulation	Rapport	Rayon de l'intervalle de confiance
1	10	21413.6	21535.2	1.005	7914.92
1	100	17437.6	17515.8	1.004	1643.9
3	10	43062.2	41481.2	0.963	7989.42
3	100	41396.8	40316.7	0.974	2827.57
une semaine	100	90644.8	90467.4	0.998	4338.6

Test de l'apport de l'approche stochastique : volatilité des prix

Ce test présente l'intérêt d'adopter l'approche stochastique pour ce problème. En effet, on a fait les étapes suivantes :

1. On a considéré deux simulateurs forward et backward avec une volatilité des prix trop faible (σ^s=$\sigma^r = 10^{-2}$). Il s'agit donc d'un cas quasi-déterministe.
2. On a résolu le problème Optimisation-Simulation en s'appuyant sur ces deux simulateurs. Grâce à la méthode StepSimulateForward, on sauvegarde, pour chaque scénario de prix, les réalisations de δ et le contrôle optimal.
3. On a implémenté une méthode *verif*[1] qui, pour un scénario de prix, prend en paramètre un contrôle et une réalisation de δ sur toute la journée et calcule la valeur du gain journalier.
4. On a appliqué la méthode *verif* avec le contrôle optimal dégagé du cas quasi-déterministe à un cas stochastique (volatilité plus importante, σ^s=σ^r=0.94).
5. Finalement, on a comparé le résultat obtenu par rapport au résultat précédent (c'est à dire le résultat obtenu en appliquant l'optimisation-simulation au cas stochastique).

Pour un seul jour et un nombre de simulations $nbSimul = 10$:

	Cas stochastique	Cas quasi-déterministe	Résultat de *verif*
Optimisation-Simulation	21413.6-21535.2	17357.1 73051	9870.61

Pour un seul jour et un nombre de simulations $nbSimul = 100$:

	Cas stochastique	Cas quasi-déterministe	Résultat de *verif*
Optimisation-Simulation	17437.6-17515.8	17269.9-31734.5	8989.93

Pour 3 jours et un nombre de simulations $nbSimul = 10$:

	Cas stochastique	Cas quasi-déterministe	Résultat de *verif*
Optimisation-Simulation	43062.2-41481.2	41323.1 38806.2	9371.73

1. verif désigne vérification

Chapitre 6

Conclusion

Dans cette étude, on a essayé d'élaborer un modèle mathématique fidèle au contexte réel du problème à résoudre "Gestion d'une batterie sur le marché spot et le marché des services systèmes". Ensuite, on a passé à la résolution en exploitant les outils mathématiques nécessaires (Programmation dynamique, régression, interpolation $\cdots$). Enfin, on a implémenté les méthodes numériques à l'aide de la librairie StOpt et on a effectué quelques tests pour évaluer l'efficacité de notre approche. Le fait de travailler sur un outil tel que StOpt nécessite de comprendre les différents aspects liés au développement informatique dans une librairie opérationnelle.

Maintenant, on va s'intéresser à améliorer le travail qui a été déjà fait en essayant d'optimiser les codes implémentés dans le but de minimiser le temps d'exécution. En effet, pour certains tests, en élevant le nombre de simulations et/ou la dimension des grilles de contrôle et des états de charge, temps d'exécution augmente énormement. De plus, ce travail sera évalué en interne à EDF R&D avec une équipe qui travaille sur un projet de smart grids et énergies renouvelables.

Ce stage à EDF R&D a été pour moi une nouvelle expérience dans le domaine de la recherche. Il a constitué également une occasion de découvrir et travailler pour la première fois sur des problème de contrôle stochastique, sans oublier bien sûr, les nouvelles connaissances que j'ai acquises en C++.

Bibliographie

[1] Xiaomin Xi, Ramteen Sioshansi, Vincenzo Marano. *A Stochastic Dynamic Programming Model for Co-optimization of Distributed Energy Storage.*

[2] Hugo Gevret, Jerome Lelong, Xavier Warin. *STochastic OPTimization library in C++*, 2016.

[3] Rahul Walawalkar, Jay Apt, Rick Mancini. *Economics of electric energy storage for energy arbitrage and regulation in New York*, 2006.

[4] Enrico Telaretti, Mariano Ippolito and Luigi Dusonchet. *A Simple Operating Strategy of Small-Scale Battery Energy Storages for Energy Arbitrage under Dynamic Pricing Tariffs*, 2015.

[5] Benjamin Heymann. *Contributions mathématiques pour la régulation et l'optimisation de la production d'électricité*, 2016.

[6] Pedram Mokrian, Moff Stephen (Stanford University). *A Stochastic Programming Framework for the Valuation of Electricity Storage.*

[7] Thomas Mercier, Julien Jomaux, Emmanuel De Jaeger. *Stochastic Programming for Valuing Energy Storage Providing Primary Frequency Control.*

[8] Tiago Soares, Pierre Pinson, Senior Member, IEEE, Tue V. Jensen, Hugo Morais, Member, IEEE. *Optimal Offering Strategies for Wind Power in Energy and Primary Reserve Markets.*

[9] Guannan He, Qixin Chen, Member, IEEE, Chongqing Kang, Senior Member, IEEE, Pierre Pinson, Senior Member, IEEE, and Qing Xia, Senior Member, IEEE. *Optimal Bidding Strategy of Battery Storage in Power Markets Considering Performance-Based Regulation and Battery Cycle Life.*