



**HAL**  
open science

# Modèles d'optimisation pour la gestion de consommation d'énergie multi-flux

David Wu

► **To cite this version:**

David Wu. Modèles d'optimisation pour la gestion de consommation d'énergie multi-flux. Optimisation et contrôle [math.OC]. Sorbonne Université, 2022. Français. NNT : 2022SORUS221 . tel-03850693

**HAL Id: tel-03850693**

**<https://theses.hal.science/tel-03850693>**

Submitted on 14 Nov 2022

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



# THÈSE DE DOCTORAT DE SORBONNE UNIVERSITÉ

Spécialité INFORMATIQUE

École doctorale Informatique, Télécommunications et Électronique (Paris)

Présentée par

**David WU**

Pour obtenir le grade de

**DOCTEUR de SORBONNE UNIVERSITÉ**

---

MODÈLES D'OPTIMISATION POUR LA GESTION DE  
CONSOMMATION D'ÉNERGIE MULTI-FLUX

---

Soutenue le 20 mai 2022 devant le jury composé de :

<i>Rapporteurs :</i>	Sophie DEMASSEY	Mines ParisTech
	Pierre FOUILHOUX	Université Sorbonne Paris Nord
<i>Examineurs :</i>	Jean-Philippe GAYON	Université Clermont Auvergne
	Safia KEDAD-SIDHOUM	CNAM
	Patrice PERNY	Sorbonne Université
<i>Directeur :</i>	Viet Hung NGUYEN	Université Clermont Auvergne
<i>Encadrants :</i>	Michel MINOUX	Sorbonne Université
	Hai TRAN	Energisme



# Remerciements

Cette thèse est un grand travail personnel, mais c'est aussi un travail impossible à réaliser seul. J'ai reçu le soutien, les conseils et les encouragements de nombreuses de personnes, et je souhaite exprimer ma plus grande gratitude à tout mon entourage. Ce travail n'aurait pu aboutir sans votre concours à toutes et à tous.

Je souhaite commencer en remerciant mon directeur de thèse, le Professeur Viet Hung Nguyen, et mon co-directeur de thèse, le Professeur Michel Minoux, qui m'ont accompagné tout au long de cette aventure. J'ai vraiment beaucoup appris auprès de vous. Vos aides et conseils ont été très formateur. Je souhaite ensuite remercier les membres de mon comité de suivi et mon jury de thèse.

Je remercie également la direction d'Energisme, Thierry Chambon, Pierre Vidal ainsi que Hai Tran. Je remercie Daniel Boutrin, Long Do et Trang Tran de m'avoir conseillé et accompagné dans mes recherches. Je souhaite remercier également les personnes suivantes à Energisme : Alexandre Mertz, Cathy Barbosa, Simon Castres-Staint-Martin, Aurélien Barbier, Victor Sicot, Maxime Blanloeil, Josselin Auger, Guillaume Delrieu, Farouk Imani, Hoang Nguyen, Jérémie Bosom, Loup-Noé Levy et Soline Aury.

Je remercie mes camarades de Master pour les excellents moments passés ensemble : Nicolas De Bufala, Tom Portoleau, Malcolm Auffrey et Orso Negroni. Je remercie aussi mes camarades de bureau au LIP6 : Gaspard Ducamp, Marvin Lasserre, Clara Charon et Cassandre Leroy. Je remercie Alexandra You pour la relecture du manuscrit.

Je souhaite terminer la liste en remerciant toute ma famille et mes amis pour l'aide, le soutien et la sympathie inestimable. Je voudrais qu'ils acceptent l'expression de tout mon respect, ma gratitude et mon amitié.

Un grand merci à vous tous!!!

## *REMERCIEMENTS*

---

# Table des matières

Remerciements	I
Table des figures	VIII
Liste des tableaux	X
Liste des algorithmes	XI
<b>1 Introduction</b>	<b>1</b>
1.1 Introduction générale . . . . .	2
1.1.1 Transition écologique et transition énergétique . . . . .	2
1.1.2 Microgrid et smartgrid . . . . .	3
1.1.3 Internet of Things et Big Data . . . . .	4
1.1.4 Gestion des données récoltées . . . . .	5
1.2 Questions de recherche . . . . .	5
1.3 Préliminaires . . . . .	6
1.3.1 Série temporelle . . . . .	6
1.3.2 Programmation linéaire en nombres entiers . . . . .	9
1.3.3 Optimisation convexe . . . . .	10
1.3.4 Optimisation robuste . . . . .	11
1.4 Organisation et contributions de la thèse . . . . .	12
1.4.1 Algorithme efficace pour l'optimisation de la puissance sous- crite des contrats . . . . .	12
1.4.2 Modèle mathématique pour le problème de marnage de type RFLATS . . . . .	13
1.4.3 Solution robuste pour le problème de marnage de type RFLATS . . . . .	13

## **I Problème d'optimisation de la souscription de contrat** **15**

<b>2 Modèle pour le PCSE</b>	<b>17</b>
2.1 Introduction . . . . .	18

TABLE DES MATIÈRES

---

2.2	Contexte de PSCE . . . . .	19
2.2.1	ENEDIS et les principes de tarification . . . . .	19
2.2.2	Structure tarifaire . . . . .	20
2.2.3	Domaine de tension BTinf . . . . .	22
2.2.4	Domaine de tension BTsup . . . . .	23
2.2.5	Domaine de tension HTA . . . . .	23
2.3	Spécificité du problème . . . . .	27
2.3.1	Problème de flot de coût convexe à objectif séparable mi- nimum . . . . .	27
2.3.2	Problème de régression isotonique . . . . .	28
2.3.3	Problème convexe à objectif séparable sous des contraintes d'ordre total . . . . .	29
2.3.4	Le PCSE face au Big Data . . . . .	29
2.4	Industrialisation du PSCE . . . . .	31
2.5	Résumé du chapitre . . . . .	33
<b>3</b>	<b>Algorithme pour le PSCE</b>	<b>35</b>
3.1	OPTIM_SP : un algorithme ajustable . . . . .	36
3.1.1	Présentation générale de l'algorithme . . . . .	37
3.1.2	Recherche par dichotomie ajustable . . . . .	37
3.1.3	Résoudre $\mathcal{P}^=(\emptyset)$ . . . . .	41
3.1.4	Résoudre $\mathcal{P}^=(B)$ avec $B \neq \emptyset$ . . . . .	41
3.1.5	Validité de OPTIM_SP . . . . .	41
3.1.6	Complexité de OPTIM_SP . . . . .	42
3.2	Résultats expérimentaux . . . . .	43
3.2.1	Jeux de données et solutions optimales . . . . .	43
3.2.2	Calcul de complexité de OPTIM_SP et scaling_PAV . . . . .	46
3.2.3	Évaluation comparative de OPTIM_SP et scaling_PAV . . . . .	46
3.3	Résumé du chapitre . . . . .	49
<b>4</b>	<b>PSCE avec incertitudes sur les consommations</b>	<b>51</b>
4.1	Bref rappel du problème . . . . .	52
4.2	Modèle robuste du PSCE . . . . .	53
4.2.1	Ensemble d'incertitude . . . . .	53
4.2.2	Formulation . . . . .	54
4.3	Résolution de la version robuste du PSCE . . . . .	56
4.4	Résultats expérimentaux . . . . .	58
4.4.1	Analyse de l'ensemble d'incertitude . . . . .	58
4.4.2	Résultats numériques concernant la version robuste . . . . .	59
4.4.3	Évaluation comparative de OPTIM_SP et scaling_PAV . . . . .	61
4.5	Résumé du chapitre . . . . .	61

<b>II</b>	<b>Problème de gestion optimale des réseaux d'eau</b>	<b>65</b>
<b>5</b>	<b>Modèle RFLATS pour les réseaux d'eau</b>	<b>67</b>
5.1	Introduction . . . . .	68
5.2	État de l'art autour des niveaux de marnage . . . . .	71
5.2.1	Marnages fixes . . . . .	71
5.2.2	Marnages fixes avec différenciation temporelle . . . . .	72
5.2.3	Marnages variants avec différenciation temporelle . . . . .	73
5.2.4	Marnages fixes avec des classes temporelles supplémentaires . . . . .	74
5.3	Contexte et motivations . . . . .	75
5.3.1	Contexte et cadre du problème . . . . .	75
5.3.2	Anatomie et points de passage du niveau d'eau de la solution optimale par RFLATS . . . . .	76
5.3.3	Complexité de RFLATS . . . . .	77
5.4	Formulation du problème de pompage . . . . .	81
5.4.1	Contraintes du réseau d'eau . . . . .	81
5.4.2	Contraintes de marnage fixe . . . . .	83
5.4.3	Contraintes correspondants aux périodes artificielles . . . . .	84
5.5	Résultats expérimentaux . . . . .	87
5.5.1	Instance de petit SDE rural . . . . .	87
5.5.2	Point de passage et robustesse de RFLATS . . . . .	88
5.5.3	Journée type : jour de semaine . . . . .	91
5.5.4	Journée type : dimanche . . . . .	95
5.5.5	Marnage reconstruit . . . . .	96
5.6	Résumé du chapitre . . . . .	99
<b>6</b>	<b>RFLATS avec des incertitudes sur les demandes</b>	<b>101</b>
6.1	Formulation robuste pour RFLATS . . . . .	102
6.1.1	Ensemble d'incertitude . . . . .	102
6.1.2	Formulation du problème robuste . . . . .	103
6.2	Résolution de la version robuste de RFLATS . . . . .	105
6.2.1	Détermination des marnages horaires . . . . .	105
6.2.2	Évaluation d'un marnage horaire donné . . . . .	106
6.2.3	Graphe de plus long chemin pour la recherche de la solution pire cas . . . . .	107
6.2.4	Réduction du graphe de la recherche du pire cas . . . . .	108
6.2.5	Simplification du graphe de la recherche du pire cas . . . . .	110
6.2.6	Algorithme pour la solution robuste . . . . .	112
6.3	Résultats expérimentaux . . . . .	113
6.3.1	Instance de petit SDE rural . . . . .	113
6.3.2	Résultats numériques pour la recherche du pire cas . . . . .	114
6.3.3	Résultats expérimentaux par résolution du problème robuste . . . . .	119
6.3.4	Résultats expérimentaux pour le marnage reconstruit et prix de la robustesse . . . . .	122

*TABLE DES MATIÈRES*

---

6.4 Résumé du chapitre . . . . .	124
<b>Conclusion générale et perspectives</b>	<b>127</b>
<b>Bibliographie</b>	<b>133</b>

# Table des figures

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Décomposition d'une série temporelle avec la STL, tirée de [31] . . .	8
<b>2</b>	<b>Modèle pour le PCSE</b>	<b>17</b>
2.1	Fonction objectif pour la catégorie $i$ : $F_i(x_i)$ . . . . .	25
2.2	Widget de l'optimisation de la puissance souscrite monosite . . . . .	31
2.3	Widget de l'optimisation de la puissance souscrite multisite . . . . .	32
2.4	Widget du pareto des économies potentielles par site . . . . .	32
<b>3</b>	<b>Algorithme pour le PSCE</b>	<b>35</b>
3.1	Histogrammes et courbes montrant la dépendance de $N_i^+(x_i)$ par rapport à $x_i$ pour trois exemples qui font partie des ensembles de données D1 et D5 représentant les puissances appelées décrits dans la Section 3.2 . . . . .	39
<b>5</b>	<b>Modèle RFLATS pour les réseaux d'eau</b>	<b>67</b>
5.1	Profil du niveau d'eau avec PSP . . . . .	69
5.2	Profil de marnage par FTL . . . . .	71
5.3	Profil de marnage par RFTL . . . . .	72
5.4	Profil de marnage par VTL . . . . .	74
5.5	Profil de marnage par RFLATS . . . . .	75
5.6	Différentes solutions de marnage de type RFLATS . . . . .	78
5.7	Plus court chemin pour le marnage optimal durant les heures pleines	80
5.8	Solution de RFLATS sur des instances modifiées . . . . .	89
5.9	Demande pour la journée type : jour de semaine . . . . .	91
5.10	Solution optimale pour la journée type : jour de semaine . . . . .	92
5.11	Demande pour la journée type : dimanche . . . . .	95

TABLE DES FIGURES

---

5.12	Demande pour la semaine . . . . .	96
5.13	Solutions optimales pour la semaine . . . . .	97
<b>6</b>	<b>RFLATS avec des incertitudes sur les demandes</b>	<b>101</b>
6.1	Non-convexité de la fonction de coût du problème déterministe . .	104
6.2	Exemple de plus long chemin pour le pire cas de RFLATS . . . .	108
6.3	Importance de la déviation à $t = T_{\mathbf{P}}$ . . . . .	110
6.4	Évolution du niveau pour un marnage serré sur 24 heures . . . . .	116
6.5	Évolution du niveau pour un marnage neutre sur 24 heures . . . . .	117
6.6	Évolution du niveau pour un marnage lâche sur 24 heures . . . . .	118
6.7	Évolution du niveau pour la solution déterministe sur 24 heures .	120
6.8	Évolution du niveau pour la solution robuste sur 24 heures . . . .	120
6.9	Évolution du niveau pour la solution déterministe reconstruit sur 6 jours . . . . .	123
6.10	Évolution du niveau pour la solution robuste reconstruit sur 6 jours	123

# Liste des tableaux

<b>2</b>	<b>Modèle pour le PCSE</b>	<b>17</b>
2.1	Composantes du TURPE5 . . . . .	20
<b>3</b>	<b>Algorithme pour le PSCE</b>	<b>35</b>
3.1	Caractéristiques principales pour les jeux de données $D_1$ à $D_5$ . . .	44
3.2	Résultats d'exécution pour les jeux de données $D_1, \dots, D_5$ . . . . .	47
<b>4</b>	<b>PSCE avec incertitudes sur les consommations</b>	<b>51</b>
4.1	Caractéristiques principales de $D_1$ pour le problème robuste . . . . .	58
4.2	Comparaison des coûts de diverses solutions pour l'ensemble de données $D_1$ . . . . .	59
4.3	Résultats d'exécution pour la version robuste du PSCE pour les jeux de données $D_1, \dots, D_5$ . . . . .	62
<b>5</b>	<b>Modèle RFLATS pour les réseaux d'eau</b>	<b>67</b>
5.1	Résultats pour la solution de référence REF et les solutions optimales obtenues avec PSP, FTL, RFTL et RFLATS sur la journée type : jours de semaine . . . . .	93
5.2	Résultats pour la solution de référence REF et pour les solutions optimales obtenues avec PSP, FTL, RFTL et RFLATS sur la journée type : dimanche . . . . .	95
5.3	Résultats pour la solution de référence REF et pour les solutions obtenues avec RFLATS et RFLATSRe sur la semaine . . . . .	98
<b>6</b>	<b>RFLATS avec des incertitudes sur les demandes</b>	<b>101</b>
6.1	Résultats expérimentaux pour la résolution du pire cas sur 24 heures	114

*LISTE DES TABLEAUX*

---

6.2	Comparatif des tailles de graphe de la solution $(\bar{v}_{\mathbf{P}'}, 554, 30, 4)$ pour les méthodes REDUCEDGRAPH et SIMPLIFIEDGRAPH . . . . .	115
6.3	Coût de la solution de référence, la solution déterministe et la solution robuste sur l'instance de 24 heures . . . . .	119
6.4	Coût de la solution déterministe et la solution robuste sur l'instance de 6 jours . . . . .	122

# Liste des algorithmes

<b>3</b>	<b>Algorithme pour le PSCE</b>	<b>35</b>
3.1	Pseudo code pour OPTIM_SP . . . . .	38
3.2	Pseudo code pour la recherche par dichotomie ajustable . . . . .	40
<b>4</b>	<b>PSCE avec incertitudes sur les consommations</b>	<b>51</b>
4.1	Code python pour l'utilisation de STL . . . . .	55
<b>6</b>	<b>RFLATS avec des incertitudes sur les demandes</b>	<b>101</b>
6.1	Déterminer les marnages d'un RFLATS . . . . .	106
6.2	Pseudo code pour l'évaluation d'un marnage . . . . .	106
6.3	Pseudo code pour la création de REDUCEDGRAPH . . . . .	111
6.4	Fonction d'énumération des marnages pour la résolution du problème robuste . . . . .	113



# Chapitre 1

## Introduction : contexte, motivations et contributions

### Sommaire

---

1.1	Introduction générale . . . . .	<b>2</b>
1.1.1	Transition écologique et transition énergétique . . . . .	2
1.1.2	Microgrid et smartgrid . . . . .	3
1.1.3	Internet of Things et Big Data . . . . .	4
1.1.4	Gestion des données récoltées . . . . .	5
1.2	Questions de recherche . . . . .	<b>5</b>
1.3	Préliminaires . . . . .	<b>6</b>
1.3.1	Série temporelle . . . . .	6
1.3.2	Programmation linéaire en nombres entiers . . . . .	9
1.3.3	Optimisation convexe . . . . .	10
1.3.4	Optimisation robuste . . . . .	11
1.4	Organisation et contributions de la thèse . . . . .	<b>12</b>
1.4.1	Algorithme efficace pour l'optimisation de la puissance souscrite des contrats . . . . .	12
1.4.2	Modèle mathématique pour le problème de marnage de type RFLATS . . . . .	13
1.4.3	Solution robuste pour le problème de marnage de type RFLATS	13

---

## 1.1 Introduction générale

L'accord de Paris est un accord mondial portant sur le réchauffement climatique, signé en 2015 durant la COP21 par 195 pays [34]. Depuis, les autorités font ainsi pression afin de diminuer les consommations d'énergie ainsi que de limiter les déchets et autres émissions carbone. Pour les particuliers et les professionnels, les économies financières induites par des dispositifs d'optimisation des dépenses énergétiques résonnent comme un argument plus convaincant que l'impact sur la planète [96]. Des acteurs majeurs du domaine de l'énergie cherchent de nouveaux relais de croissance, et font le choix de se rapprocher des éditeurs de logiciel dédiés à l'efficacité énergétique, comme Energisme [40]. Un tel rapprochement permet ainsi aux clients de se lancer dans leur transition énergétique à moindre coût.

### 1.1.1 Transition écologique et transition énergétique

En 2010, Rob Hopkins regroupe dans un manuel sur la transition écologique un ensemble d'idées sur les problématiques de résilience collective, d'économie circulaire et de réduction des émissions de CO<sub>2</sub> [55]. La résilience collective réagit de manière proactive aux situations difficiles. La communauté résiliente, quant à elle, se prépare aux changements économiques, sociaux et environnementaux et parviennent à de meilleurs résultats face aux crises et aux défis. L'économie circulaire, enfin, est un modèle économique à vision systémique et vise à changer de paradigme par rapport à l'économie dite linéaire : elle comprend les notions d'économie verte, d'économie de l'usage ou de l'économie de la fonctionnalité, de l'économie de la performance et de l'écologie industrielle, permettant d'augmenter l'efficacité à tous les stades de l'économie des produits.

La transition énergétique désigne une modification structurelle profonde et durable de tous les secteurs de l'énergie, de la production à la consommation finale et visant à limiter les émissions de gaz à effet de serre. Elle résulte des évolutions techniques, des prix et de la disponibilité des ressources énergétiques, en abandonnant les combustibles fossiles au profit d'énergies renouvelables (solaire, éolien, hydraulique, bois-énergie...), mais aussi d'une volonté de réduire les effets négatifs de chacun des secteurs sur l'environnement. La transition énergétique est aussi une transition comportementale et sociotechnique [23, 47], qui implique une modification radicale de la politique énergétique par l'application de la sobriété énergétique et l'efficacité énergétique. La sobriété énergétique vise

la diminution des consommations d'énergies par des changements de modes de vie et des transformations sociétales par une approche non-techniciste, centrée sur les comportements, l'organisation et la structure de la société. Elle se manifeste par la limitation des biens et services produits et consommés à un niveau suffisant [101]. L'efficacité énergétique désigne, pour un service rendu identique, l'état de fonctionnement d'un système pour lequel la consommation d'énergie est minimisée, et s'appuie généralement sur l'optimisation des consommations.

Dans les enjeux de transition énergétique et de réduction des consommations, le secteur du bâtiment joue un rôle primordial. En effet, en 2019, les bâtiments des secteurs résidentiels et tertiaires du territoire métropolitain français représentent 45% de la consommation finale d'énergie et 21% des émissions directes de CO<sub>2</sub> [30]. En France, le décret tertiaire, entré en vigueur en octobre 2019, oblige tous les propriétaires et bailleurs de bâtiments tertiaires privés et publics de plus de 1000m<sup>2</sup> à réduire leurs consommations d'énergie à court et moyen terme. Les objectifs de réduction sont fixés à -40% en 2030, -50% en 2040 et -60% en 2050, par rapport à 2010. Par exemple, les réseaux utilisant de l'électricité peuvent récupérer et utiliser leurs données de consommation afin de mieux gérer leurs consommations ou de souscrire à des contrats plus adaptés.

### 1.1.2 Microgrid et smartgrid

Un réseau électrique (ou *grid*) est défini comme un système de fourniture d'électricité à des consommateurs. Ce système est principalement composé de trois entités : la production, l'acheminement et la charge (la demande). Un microgrid peut produire sa propre électricité pour répondre à une partie ou toute sa consommation à un moment donné. Dans ce cas, les unités de production sont installées à proximité de la charge. Lorsqu'un microgrid est connecté au réseau principal afin d'échanger de l'électricité avec lui, on dit que c'est un microgrid connecté, et lorsqu'il est indépendant du réseau principal, on dit qu'il est autonome [82].

Un smartgrid est un microgrid qui favorise la circulation d'informations entre la partie des entreprises de service public, la partie production locale et la partie consommateur par une série de mesures faites par des compteurs intelligents ou compteurs communicants [42]. Ces mesures permettent d'ajuster le flux d'électricité en temps réel en adaptant la consommation aux capacités instantanées de production. On peut, par exemple, décaler certaines consommations en dehors

des heures de pointe afin de réduire les pics de consommation. L'émergence de moyens de stockage de l'électricité offre de nouvelles possibilités au smartgrid. Le stockage de l'énergie et le contrôle de la production et de la distribution d'électricité sont des aspects importants du réseau intelligent [58]. Le smartgrid est aussi présenté comme un moyen efficace dans la lutte contre le réchauffement climatique.

Grâce à la grande quantité de données récupérées, le smartgrid soulève de nombreux problèmes de gestion et d'optimisation entre le réseau national (*utilities*), la production/stockage (*supply-side*) et la demande (*demand-side*) [96]. Depuis 2014, de très nombreux travaux ont été menés sur l'optimisation du smartgrid [80, 99]. Les *utilities* interviennent principalement autour des problèmes d'acheminement liés à la souscription de contrats d'électricité [98, 106]. Le *supply-side* optimise autour de l'intégration des batteries [62], de la génération d'énergie [57] et des coûts environnementaux [109]. Le *demand-side management* (DSM) est étudié autour des modèles génériques [49, 68], de la gestion automatique [91], de l'application du *load-shifting* [50, 71, 72], de l'utilisation des *feedbacks* [43], ou de l'optimisation de l'énergie domestique [33, 59].

### 1.1.3 Internet of Things et Big Data

L'Internet des objets ou *Internet of Things* (IoT) est défini comme étant un réseau d'objets inter-connectés [8]. Il désigne un nombre important d'objets connectés à Internet permettant ainsi une communication entre les objets physiques et numériques, par le biais de l'utilisation de nombreux protocoles de communication. Le premier concept apparaît vers les années 90 avec l'*internet des machines* (les ordinateurs en réseau) [83], puis avec les *objets connectés* grâce à l'apparition d'une technologie de communication sans fil, la Radio Frequency Identification (RFID). Les générations suivantes utilisent de nombreux protocoles de communication tels que Bluetooth, Zigbee, WiFi, Sigfox et LoRa [41, 74]. Le nombre d'objets connectés (réfrigérateurs, machines à café, brosses à dents, téléphones et appareils intelligents ...) est estimé à entre 28 milliards [41] et 50 milliards en 2020 [74], puis de 150 milliards en 2025 [52].

La collecte de données énergétiques est devenue une actrice de la transition écologique et un enjeu majeur dans la course à la sauvegarde de la planète. L'apparition des capteurs communicants permet de répondre à un besoin de mesurer

l'évolution des paramètres environnementaux et comportementaux dans des milieux naturels [93], militaires [94], agricoles [76], médicaux [104], industriels ou civils [24]. Ces nouvelles formes de connexion permettent de rassembler de nouvelles masses de données sur le réseau. L'IoT est en effet en partie responsable de l'accroissement exponentiel du volume de données générées sur le réseau, à l'origine du *Big Data* et offrant donc de nouvelles connaissances et nouvelles formes de connaissances.

#### 1.1.4 Gestion des données récoltées

Les données de consommation sont souvent représentées comme un processus stochastique représentant l'évolution de la consommation et basé sur des facteurs d'influence. Ces facteurs d'influence sont en général les conditions extérieures (température, climat...), les caractéristiques physiques du logement (type, âge...), les appareils et les occupants (occupation et comportement...), les prix et les abonnements [10, 66, 110]. Les travaux sur l'énergie d'éclairage dans les grands immeubles de bureaux montrent que la consommation peut être simulée avec précision en tenant compte du comportement des occupants et des variations saisonnières, et qu'elle peut être décrite en utilisant par exemple, la distribution de Poisson et la distribution normale [110]. Le modèle de consommation est utile pour prévoir la consommation à court terme, à moyen terme, à long terme [100] et à très long terme [10]. De nombreux travaux reposent sur une modélisation précise de chaque appareil du foyer [61, 86, 90]. Les auteurs de [35] présentent une approche basée sur les chaînes de Markov cachées pour la modélisation et l'analyse statistique des courbes de consommation électrique.

## 1.2 Questions de recherche

Cette thèse s'inscrit dans une convention CIFRE entre Energisme et le Laboratoire d'Informatique de Paris 6 (LIP6). Energisme est une entreprise spécialisée dans le développement d'une plateforme logicielle dédiée à la performance énergétique et environnementale. La société propose une solution destinée à collecter, à analyser et à traiter les données énergétiques des entreprises et des collectivités (électricité, gaz, eau, air comprimé...) et à fournir des tableaux de bord ainsi que des outils d'aide à la décision.

La très grande quantité de données recueillies et la diversité des utilisateurs de la plateforme fournissent un cadre parfait pour des études d'optimisation énergétique. Parmi elles, on peut citer l'optimisation de la souscription de contrat d'électricité et la gestion optimale des réseaux d'eau qui sont les problèmes considérés dans cette thèse.

**Optimisation de la souscription de contrat** Ce problème consiste à trouver les meilleurs paramètres d'un contrat d'électricité pour un client en fonction de son historique de consommation d'électricité sur une période de temps fixe. L'objectif est d'optimiser la facture d'électricité composée du coût de la fourniture, du coût de la souscription du contrat d'électricité, des pénalités dues aux dépassements de puissance souscrite et de certains coûts fixes. Ce problème est formulé comme la minimisation d'une fonction convexe à objectif séparable soumise à des contraintes d'ordre total. Le problème est d'abord étudié dans sa version déterministe (cf. [Chapitre 2](#) et [Chapitre 3](#)) puis dans une version robuste où les données de consommation sont soumises à un ensemble d'incertitudes (cf. [Chapitre 4](#)).

**Gestion optimale des réseaux d'eau** La gestion optimale d'un réseau d'eau consiste à minimiser le coût énergétique généré par les opérations de pompage pour alimenter un ou plusieurs réservoir(s) d'eau surélevé(s) à partir d'une source. Les pompes se mettent automatiquement en marche ou à l'arrêt lorsque le niveau d'eau atteint une valeur donnée dans le château d'eau. On appelle ces niveaux des "niveaux de marnages" ou simplement "marnages". Le problème est aussi étudié dans sa version déterministe avec la programmation linéaire en nombres entiers (cf [Chapitre 5](#)), puis dans sa version robuste où les données de consommation sont soumises à un ensemble d'incertitudes (cf. [Chapitre 6](#)).

## 1.3 Préliminaires

Cette section présente quelques concepts techniques présents dans cette thèse, ainsi que les approches que nous avons choisi d'appliquer.

### 1.3.1 Série temporelle

Une série temporelle (*time series*) est une série de points de données indexées dans l'ordre chronologique et représentant l'évolution d'une quantité spécifique

au cours du temps. Les séries temporelles permettent l'analyse de comportement, que ce soit son évolution passée ou son évolution future. Ainsi, les techniques et les concepts de probabilité et de statistique sont souvent utilisés pour décrire les phénomènes.

Dans cette thèse, les données de consommation brutes récoltées depuis les capteurs sont nettoyées et normalisées. Pour ce faire, les données sont considérées comme une série temporelle, puis on applique la méthodologie *Seasonal and Trend decomposition using Loess* (STL) de Cleveland, R. B., Cleveland, W. S., McRae, J. E., et Terpenning, I. (1990) [31]. C'est une procédure de filtrage qui permet de décomposer une série temporelle en composante de tendance, en composante de saisonnière et composante résiduelle :

$$rawdata = trend + seasonal + remainder,$$

où  $rawdata(t)$  est la donnée brute.

La Figure 1.1 (tirée de [31]) montre la décomposition STL de mesures atmosphériques de CO<sub>2</sub> moyennes d'Avril 1974 à Décembre 1986.

La composante *trend* représente la variation à basse fréquence des données, comme les changements non stationnaires à long terme, e.g. l'augmentation progressive de la quantité de CO<sub>2</sub> au fil du temps. La composante *seasonal* représente la variation périodique de la donnée, e.g. considérant que la quantité de CO<sub>2</sub> augmente les étés et diminue les hivers de chaque année, où la période est donnée en input. La composante *remainder*, ou résiduelle, représente la variation restante en dehors des composantes de tendance et de saisonnalité. Une bonne méthode de décomposition aura un résiduel avec certaines propriétés comme l'absence de corrélation et une moyenne nulle, et certaines propriétés facultatives comme une variance constante et une distribution normale. Notons que la Figure 1.1 présente des données manquantes (visible par les trous de données dans *Data*) et des données aberrantes (visible par les grands écarts de données dans *Data* et donc dans *Remainder*).

STL a une conception simple qui consiste en une séquence d'applications du lisseur de *Loess* ; cette simplicité permet d'analyser les propriétés de la procédure et de la calculer rapidement. Cette méthodologie STL satisfait les critères suivants :

- une conception simple et facilement utilisable,

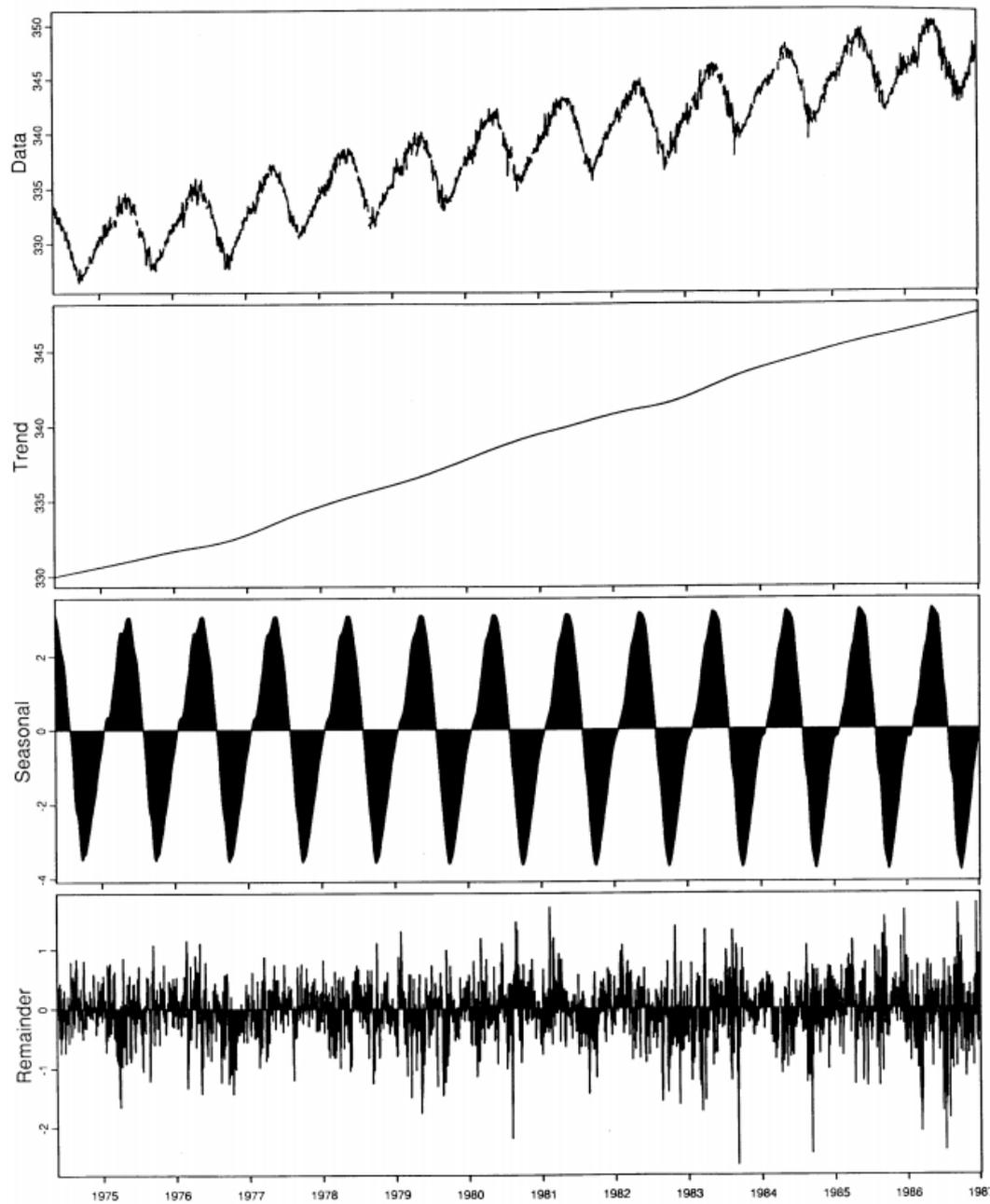


FIGURE 1.1 – Décomposition d’une série temporelle avec la STL, tirée de [31]

- la flexibilité dans la spécification des montants de variation de la tendance et des composantes saisonnières,
- la période de la composante saisonnière est un entier supérieur à 1,
- la possibilité de gérer les valeurs manquantes,
- la robustesse face aux données aberrantes,
- la facilité d'implémentation et calcul rapide.

La méthodologie STL présente quelques inconvénients. En particulier, elle ne traite pas automatiquement les variations de calendrier (e.g. les 29 Février ou les changements d'horaire), et elle ne fournit que des facilités pour les décompositions additives.

**Loess** La régression (polynomiale) locale ou régression mobile est une généralisation de la moyenne mobile et de la régression polynomiale. Ses méthodes les plus courantes sont *LOcally Estimated Scatterplot Smoothing (LOESS)* et *LOcally WEighted Scatterplot Smoothing (LOWESS)*. Ce sont des méthodes de régression non paramétrique qui combine plusieurs modèles de régression dans un méta-modèle basé sur la méthode des k plus proches voisins.

Par la suite, les données qu'on utilise sont collectées toutes les 10 minutes pour le problème de souscription de contrat et toutes les 15 minutes pour le problème de gestion des systèmes de distribution d'eau, puis nous considérons que la consommation nettoyée est égale à  $trend + seasonal$ . Notons que le résiduel pourra être utilisé pour déterminer des variations plausibles autour de la consommation.

### 1.3.2 Programmation linéaire en nombres entiers

La programmation linéaire en nombres entiers (PLNE) (ou optimisation linéaire en nombres entiers) est un domaine de la programmation mathématique. Un programme est dit linéaire lorsque les contraintes et la fonction de coût peuvent être exprimées par des combinaisons linéaires des variables. Un programme est dit en nombres entiers lorsque certaines de ces variables ne sont autorisées qu'à prendre des valeurs dans l'ensemble des entiers. Il y a deux raisons principales pour l'utilisation de variables entières lors de la modélisation de problèmes. La première est lorsque les variables représentent des quantités entières

(e.g. 1 machine, 2 machines...). La deuxième est lorsque les variables représentent des décisions binaires (e.g. activer ou ne pas activer). La programmation linéaire en nombres entiers peut donc être utilisée dans de nombreux domaines d'application de recherche opérationnelle comme dans : la planification, l'ordonnancement, le partitionnement...

La programmation linéaire en nombres entiers est généralement un problème NP-difficile [46]. La complexité est montrée polynomiale quand le nombre de variables est fixé [70].

Les méthodes de résolution exactes connues sont : la méthode des plans sécants (*cutting plane method*), le principe de séparation et évaluation (*branch and bound*), ou la combinaison des deux méthodes (*branch and cut*) etc... La contrainte d'intégrité est la contrainte qui force les variables à prendre des valeurs entières. Dans les méthodes citées précédemment, on fait appel à la résolution de la relaxation linéaire du problème, c'est à dire ce même problème sans la contrainte d'intégrité. On appelle le gap d'intégralité (*integrality gap*), le ratio maximum entre la valeur optimale du problème en nombres entiers et celle de sa relaxation linéaire. La résolution de PLNE cherche souvent à trouver des solutions améliorants ce gap d'intégrité en ajoutant des inégalités valides, c'est à dire en ajoutant des inégalités toujours vraies. Il existe des solvers, commerciaux ou non, adoptant cette approche pour résoudre les PLNE, dont le solver commercial Gurobi dans sa version 9.0.2, que nous utilisons dans cette thèse.

### 1.3.3 Optimisation convexe

L'optimisation convexe est un domaine de l'optimisation mathématique qui étudie la minimisation des fonctions convexes sur des ensembles convexes. La programmation linéaire est un cas particulier de l'optimisation convexe. Les problèmes d'optimisation convexes ont des propriétés intéressantes telles que tout optimum local est aussi un optimum global. Certaines classes de problèmes d'optimisation convexe admettent des algorithmes en temps polynomial, comme par exemple la programmation linéaire.

Les problèmes d'optimisation peuvent être résolus par la méthode des plans sécants, la méthode de l'ellipsoïde (*ellipsoid method*) ou la méthode de sousgradient (*subgradient method*) ou la méthode de points intérieurs (*interior-point method*) est proposée [81]. Dans cette thèse, les problèmes convexes étudiés sont à objectif

séparables, c'est à dire qui s'écrivent comme une somme de fonctions, et permettent de se ramener à la solution de sous-problèmes convexes sans contrainte, qui peuvent être facilement résolus par la descente du gradient (*gradient descent*) ou la méthode de Newton (*Newton's method*) ou la recherche par dichotomie (*binary search*). Nous utilisons la recherche par dichotomie afin de trouver le minimum d'une fonction à une variable qui est convexe et non différentiable dans un intervalle donné, où à chaque itération on évalue un certain point puis on décide de garder la partie gauche ou droite de l'intervalle de recherche. Le point évalué est souvent le point médian de l'intervalle, mais nous montrerons dans le 3 que certains autres points peuvent parfois être plus intéressants à choisir.

### 1.3.4 Optimisation robuste

Les problèmes d'optimisation peuvent faire intervenir des paramètres incertains. Les incertitudes peuvent provenir des erreurs de mesure ou de prévision ou d'estimation, ou aussi d'arrondis sur les calculs, ainsi ces écarts peuvent produire des solutions très mauvaises voire même non réalisables.

L'objectif de l'optimisation stochastique est l'optimisation d'une certaine espérance [19, 32]. On cherche à trouver des solutions bonnes en moyennes, il est donc nécessaire d'avoir des statistiques précises sur les paramètres incertains. L'objectif de l'optimisation robuste est l'optimisation d'un pire cas [12], où on cherche à se protéger contre des événements incertains néfastes. L'approche d'optimisation robuste peut surpasser de manière significative l'approche d'optimisation stochastique lorsqu'on s'intéresse à de faibles niveaux de risque [79].

L'ensemble d'incertitude ellipsoïdal est étudié par El Ghaoui, L., *et al.* (1997, 1998) [38, 39] et par Ben-Tal, A., et Nemirovski, A. (1999) [11]. Les paramètres incertains  $a_{ij}$  prennent la valeur  $\bar{a}_{ij} + \hat{a}_{ij}\xi_i$ , tel que  $\bar{a}_{ij}$  est la valeur nominale,  $\hat{a}_{ij}$  est la valeur maximale de la déviation et  $\xi_i$  est dans  $[-1, 1]$ . La contrepartie robuste est équivalente à un SOCP (*Second-Order Cone Program*). Un paramètre  $\Omega_j$  contrôle la robustesse en excluant les valeurs extrêmes qui sont peu probables, et implique que la solution robuste satisfait la contrainte  $j$  avec une probabilité d'au moins  $1 - e^{-\Omega_j^2/2}$  [13].

L'ensemble d'incertitude polyédrale est étudié par Ben-Tal, A., et Nemirovski, A. (1999) [11]. L'ensemble des incertitudes réalisables peut être exprimé par un polyèdre, la contrepartie robuste est équivalente à un problème d'optimisation linéaire. La taille du problème croît de façon polynomiale en fonction de la

taille du problème déterministe et des dimensions de l'ensemble des incertitudes.

L'ensemble d'incertitude de type budget est une famille d'ensemble d'incertitude polyédrale étudiée par Bertsimas, D., et Sim, M. (2004) [15]. Les paramètres incertains  $a_{ij}$  prennent cette fois une valeur dans l'intervalle  $[\bar{a}_{ij} - \hat{a}_{ij}, \bar{a}_{ij} + \hat{a}_{ij}]$ , et où un budget d'incertitude  $\Gamma_i$  contrôle le nombre de paramètres autorisés à dévier de leurs valeurs nominales et implique que la solution robuste satisfait la contrainte  $i$  avec une probabilité d'au moins  $e^{-\frac{\Gamma_i^2}{2|\mathcal{J}_i|}}$ . Le budget d'incertitude contrôle le compromis entre l'optimalité de la solution et sa robustesse par rapport aux perturbations des paramètres incertains. Dans cette thèse, on considère que le *remainder* de la décomposition STL est une réalisation possible de l'ensemble d'incertitude. Ainsi le choix des valeurs de la déviation maximale  $\hat{a}_{ij}$  et de budget d'incertitude  $\Gamma_i$  dépendront de *remainder* (e.g. son étendu, son écart-type ...).

## 1.4 Organisation et contributions de la thèse

Les résultats obtenus sur les problèmes d'optimisation énergétiques étudiés dans cette thèse comportent à la fois des contributions scientifiques et des innovations en termes d'applications industrielles.

### 1.4.1 Algorithme efficace pour l'optimisation de la puissance souscrite des contrats

Le contexte du problème d'optimisation de la puissance souscrite des contrats d'électricité est présenté dans le [Chapitre 2](#), et nous montrons que le problème se ramène à un problème convexe à objectif séparable sous des contraintes d'ordre total. De nos jours, les problèmes énergétiques peuvent faire intervenir un très grand nombre de données de consommation. En effet, l'augmentation du nombre de données augmente le nombre d'opérations arithmétiques, il n'est donc pas toujours correct de considérer que l'évaluation d'une solution se fasse en temps constant. Ainsi, l'effort de calcul est donc à reconsidérer.

Dans le [Chapitre 3](#), nous présentons l'algorithme OPTIM\_SP qui est spécialement conçu pour réduire l'effort de calcul sur des données historiques à grande échelle du problème de souscription de contrat d'électricité, en se basant sur l'utilisation d'une recherche par dichotomie asymétrique et ajustable. En effet,

l'analyse de la formulation du problème révèle que le nombre d'opérations arithmétiques nécessaire pour évaluer une solution dépend fortement du nombre de données supérieures aux éléments de la solution. Pour ce problème, OPTIM\_SP s'avérera être plus efficace que scaling\_PAV de Ahuja, R. K., et Orlin, J. B. (2001) [3], l'un des algorithmes les plus efficaces pour résoudre les problèmes convexes à objectif séparables.

Dans le [Chapitre 4](#), nous présentons une version robuste du problème de souscription de contrat d'électricité, où nous utilisons une combinaison de la décomposition STL de [31] et de l'ensemble d'incertitude de type budget de [15]. Il est montré que le problème se résout aussi en utilisant OPTIM\_SP, et où la détermination du pire cas à chaque évaluation de la fonction objectif se fait par la résolution d'un problème de sac à dos à choix multiple.

Ces travaux sur la version déterministe et sur la version robuste du problème sont exposés dans [106], présentés à WGCO2019.

### 1.4.2 Modèle mathématique pour le problème de marnage de type RFLATS

Le problème de gestion d'un système de distribution d'eau est largement étudié sous le nom de *Pump Scheduling Problem*. Nous abordons la gestion dans une version peu étudiée mais très largement utilisé : les niveaux de marnages. Le marnage de type *Reduced Fixed trigger Levels in Additional Time Slots* (RFLATS) est proposé par Quintiliani, C., et Creaco, E. (2019) [88] afin d'offrir plus d'économie sur les opérations de pompage avec des niveaux marnages. Dans le [Chapitre 5](#), nous proposons un premier modèle linéaire en nombres entiers pour le problème avec RFLATS. Ces travaux ont obtenu le Best Paper Award de la conférence IEEE-RIVF2021 [105]. On remarque par la suite que la gestion RFLATS offre une certaine robustesse, et nous permet un découpage du problème en plus petits problèmes de 24 heures chacun.

### 1.4.3 Solution robuste pour le problème de marnage de type RFLATS

Dans le [Chapitre 6](#), nous définissons une version robuste du problème de marnage de type RFLATS en utilisant la même approche de robustesse que pour le problème de souscription de contrat d'électricité, c'est à dire une combinaison

de la décomposition STL de [31] et de l'ensemble d'incertitude de type budget de [15].

Les déviations permettent d'augmenter la consommation à certain moment, afin de décaler les activations des marnages et donc décaler les moments de pompage un peu plus tôt ou plus tard dans le temps. Pour un marnage donné, il est montré que le coût du pire cas peut être obtenu grâce à un algorithme de plus long chemin dans un graphe sans circuit. Une solution approchée intéressante peut être obtenue en réduisant la taille du graphe en question en choisissant des transitions stratégiques. Le problème robuste est ensuite résolu par l'énumération des marnages possibles.

## Première partie

# Problème d'optimisation de la souscription de contrat



# Chapitre 2

## Modèle pour le problème de souscription de contrat d'électricité

### Sommaire

---

2.1	Introduction . . . . .	18
2.2	Contexte de PSCE . . . . .	19
2.2.1	ENEDIS et les principes de tarification . . . . .	19
2.2.2	Structure tarifaire . . . . .	20
2.2.3	Domaine de tension BTinf . . . . .	22
2.2.4	Domaine de tension BTsup . . . . .	23
2.2.5	Domaine de tension HTA . . . . .	23
2.3	Spécificité du problème . . . . .	27
2.3.1	Problème de flot de coût convexe à objectif séparable minimum . . . . .	27
2.3.2	Problème de régression isotonique . . . . .	28
2.3.3	Problème convexe à objectif séparable sous des contraintes d'ordre total . . . . .	29
2.3.4	Le PCSE face au Big Data . . . . .	29
2.4	Industrialisation du PSCE . . . . .	31
2.5	Résumé du chapitre . . . . .	33

---

Nous abordons dans ce chapitre le problème de souscription des contrats d'électricité (PSCE), qui consiste à trouver les meilleurs paramètres d'un contrat d'électricité pour un client en fonction de son historique de consommation d'électricité sur une période de temps fixe. L'objectif est d'optimiser la facture d'électricité composée du coût de la fourniture, du coût de la souscription du contrat

d'électricité, des pénalités dues aux dépassements de puissance souscrite et de certain coût fixe. Ce problème est formulé comme la minimisation d'une fonction convexe à objectif séparable soumise à des contraintes d'ordre total.

## 2.1 Introduction

Nous nous intéressons à la minimisation de la facture d'électricité du point de vu du problème de souscription des contrats d'électricité (PSCE). Un industriel peut posséder de nombreux sites, e.g. magasins, usines, dépôts... Chacun des sites peut être relié à plusieurs points de livraison d'électricité (PDL) (parfois noté point de connexion ou point de raccordement). Le PSCE est donc une question importante pour de nombreux clients industriels qui ont un contrat d'électricité pour chacun de ses points de livraison.

Les contrats peuvent définir deux types de seuil, représentant le maximum de la consommation totale d'énergie en kilowattheures et le maximum de la demande de pointe (i.e. la demande sur une courte période) en kilowatt. Certains types de seuils peuvent prendre des valeurs différentes dans le temps (suivant la catégorie temporelle : heures pleines, heures creuses ...), et il peut y avoir des contraintes entre ces seuils afin de garantir un équilibre de la charge dans le réseau. Le dépassement de ces seuils est parfois autorisé en échange de certaines pénalités. L'objectif de PSCE est de déterminer les seuils qui minimisent la facture d'électricité sur des données historiques de consommation électrique.

Des métaheuristiques ont été proposées pour le choix des seuils d'un PSCE de consommateurs industriels, incluant les algorithmes génétiques [102], les optimisations par essais particuliers [69], les optimisations par essais de chats [60], des méthodes Taguchi avancées [107]. Dans le cas où il y a un coût fixe pour une demande ne dépassant pas le seuil et lorsqu'il y a un coût supplémentaire pour un dépassement du seuil, une formulation avec un programme linéaire peut être proposée et résolue en temps polynomial [29]. Un modèle mathématique est développé en combinant un problème de sélection du contrat d'énergie avec un problème de dimensionnement de lot en considérant les sources d'énergie renouvelables a été étudiée dans [89]. L'utilisation d'une tarification basée sur ce que l'on appelle la "puissance souscrite" (i.e. les seuils) pousse les gens à adapter leur comportement de consommation d'électricité afin de correspondre à son abonnement [98].

Dans la suite, nous étudions le PSCE, dans sa version française proposée par ENEDIS et son TURPE, utilisant des puissances souscrites et formulé comme la minimisation d'une fonction convexe à objectif séparable soumise à des contraintes d'ordre total. Cette recherche de puissance souscrite est basée sur la facturation du coût de la fourniture, du coût de la souscription du contrat d'électricité et du coût des pénalités pour les dépassements par rapport aux puissances souscrites en question. Notons que le coût de la fourniture est une constante car il n'est pas lié à la puissance souscrite.

## 2.2 Contexte de PSCE

### 2.2.1 ENEDIS et les principes de tarification

En France, ENEDIS est chargé de la gestion de 95% du réseau de distribution d'électricité. Les Tarifs d'Utilisation des Réseaux Publics d'Électricité sont appelés TURPE. Le TURPE est le tarif payé par tous les utilisateurs des réseaux, consommateurs, producteurs, gestionnaires des réseaux et fournisseurs, pour chaque contrat d'accès et pour chaque PDL, et vise à couvrir les coûts du distributeur dès lors qu'ils correspondent à ceux d'un gestionnaire de réseau efficace. Pour un client résidentiel, cela représente environ 30% de sa facture d'électricité TTC.

Ce tarif unique comporte trois composantes principales : le soutirage, la gestion de la clientèle et le comptage. Il reflète ainsi les coûts engagés par les gestionnaires des réseaux, et inclut une rémunération de leurs investissements. La tarification comprend :

- d'une part, le tarif proprement dit (barèmes pour chaque option de la grille tarifaire) et ses règles d'application,
- d'autre part, les tarifs des prestations de services qu'Enedis propose à tous les utilisateurs du réseau qui en font la demande. Ces prestations font l'objet d'un catalogue dont les prix sont publics. Il est disponible sur le site Internet d'Enedis : [www.enedis.fr/prestations](http://www.enedis.fr/prestations).

Le TURPE obéit aux règles suivantes :

- La péréquation tarifaire : Le tarif est identique sur l'ensemble du territoire national, conformément au principe d'égalité de traitement inscrit dans le Code de l'énergie,

	CG	Composante annuelle de gestion
+	CC	Composante annuelle de comptage
+	CS	Composante annuelle de soutirage
+	CMDPS	Composante mensuelle des dépassements de puissance souscrite
+	CACS	Composante annuelle des alimentations complémentaires et de secours
+	CR	Composante de regroupement
+	CER	Composante annuelle de l'énergie réactive
+	CI	Composante annuelle des injections
=	TURPE	

TABLEAU 2.1 – Composantes du TURPE5

- Le principe du « timbre-poste » : Le tarif est indépendant de la distance parcourue par l'énergie entre le point d'injection et le point de soutirage (soit entre le site producteur et le site consommateur),
- La tarification en fonction de la puissance souscrite et de l'énergie soutirée : Le tarif dépend du domaine de tension de raccordement, de la puissance souscrite et des flux physiques mesurés au(x) PDL des utilisateurs du réseau,
- L'horosaisonnalité : Les prix sont différenciés selon les saisons, les jours de la semaine et/ou les heures de la journée.

## 2.2.2 Structure tarifaire

La version appelée TURPE5bis<sup>1</sup> a été élaborée selon les principes généraux ayant fondé des précédentes versions, TURPE4 et TURPE5. En chaque PDL, le prix payé annuellement pour l'utilisation des réseaux publics de distribution est la somme des composantes dans le [Tableau 2.1](#). Les PDL sont raccordés à un domaine de tension, noté HTA, BTsup ou BTinf. Les utilisateurs doivent choisir une option tarifaire proposée pour le domaine de tension ainsi qu'une puissance souscrite, ou plusieurs puissances souscrites pour les tarifs à différenciation temporelle. L'option tarifaire détermine les coefficients dans le calcul des composantes. Selon l'utilisation, certaines composantes peuvent être égales à zéro.

---

1. [www.enedis.fr/sites/default/files/TURPE\\_5bis\\_plaquette\\_tarifaire\\_aout\\_2020.pdf](http://www.enedis.fr/sites/default/files/TURPE_5bis_plaquette_tarifaire_aout_2020.pdf)

**La composante annuelle de gestion** couvre les coûts supportés par les gestionnaires des réseaux publics de distribution pour la gestion des dossiers des utilisateurs, l'accueil physique et téléphonique, la facturation et le recouvrement. Cette composante est facturée pour chaque point de connexion sous la forme d'un terme fixe appliqué à tous les utilisateurs.

**La composante annuelle de comptage** varie selon que le dispositif de comptage est ou non propriété de l'utilisateur. Elle dépend du niveau de tension, de la puissance de soutirage souscrite et/ou de la puissance maximale d'injection.

**La composante annuelle de soutirage** comprend une part fixe, pour le coût du choix des puissances souscrites, et une part variable, pour le coût de l'acheminement de l'électricité.

**La composante mensuelle des dépassements de puissance souscrite** couvre le coût des dépassements de puissance appelée par l'utilisateur au-delà de sa puissance souscrite. ENEDIS s'efforce de répondre favorablement aux appels de puissance qui dépasseraient la puissance souscrite, à condition qu'ils ne soient pas susceptibles d'engendrer des troubles dans l'exploitation des réseaux.

**La composante annuelle des alimentations complémentaires et de secours** est facturée pour tout utilisateur bénéficiant d'une alimentation complémentaire et/ou de secours.

**La composante de regroupement** est fonction de la longueur des ouvrages des réseaux publics électriques entre chaque PDL et le PDL permettant le regroupement. En effet, les utilisateurs disposant de plusieurs points de connexion dans le domaine de tension HTA peuvent, s'ils le souhaitent, bénéficier du regroupement tarifaire pour le calcul des composantes des injections, de soutirage et des dépassements, ainsi que la composante d'énergie réactive. Dans ce cas, la facturation est établie sur la base de la somme des courbes de mesure des différents points de connexion.

**La composante annuelle de l'énergie réactive** facture l'énergie réactive soutirée pendant les mois de novembre à mars, de 6 h à 22 h, du lundi au samedi,

les jours ouvrables, pour la partie qui dépasse 40% de l'énergie active consommée pendant la même période.

**La composante annuelle des injections** est facturée pour chaque point de connexion en fonction de l'énergie active injectée sur le réseau public de distribution.

Le TURPE prévoit plusieurs domaines de tension afin de servir une tous les types de consommateurs. Dans le cadre du PSCE, les seules composantes dépendantes des puissances souscrites sont la part fixe de la CS (i.e. la partie du choix des puissances souscrites) et le CMDPS. Les autres composantes seront considérées comme des coûts fixes.

### 2.2.3 Domaine de tension BTinf

En général, un client résidentiel est raccordé en Basse Tension  $\leq 36$  (BTinf). Les options possibles sont : Courte Utilisation 4 (CU4), Courte Utilisation (CU), Moyenne Utilisation 4 (MU4), Moyenne Utilisation 2 (MU2), Longue Utilisation (LU). Plus la version d'utilisation est longue, alors plus le coût de l'abonnement (la part fixe de la CS) est élevé mais moins le prix du kWh (la part variable de la CS) est élevé, et ce en influant sur les coefficients dans les formules. Le chiffre de l'option indique le nombre de catégories temporelles utilisé parmi : heure pleine saison haute, heure creuse saison haute, heure pleine saison basse et heure creuse saison basse. En générale, les heures pleines sont de 6h00 à 22h00, et les heures creuses sont de 22h00 à 6h00, l'électricité utilisée pendant les heures pleines est facturée à un taux plus élevé que l'électricité utilisée pendant les heures creuses. La saison haute est équivalente à la période dites "d'hiver" et s'étend du mois de novembre au mois de mars, tandis que la saison basse est équivalente à la période dites "d'été" et regroupe les autres mois.

Dans le cas de BTinf, même s'il y a plusieurs catégories temporelles, il n'y a qu'une seule puissance souscrite à choisir et qui doit être inférieure ou égale à 36kVA. La CS vaut  $s * x$ , pour un coefficient  $s$  en €/kVA donné et  $x$  la puissance souscrite choisie. Aucun dépassement n'est autorisé, lorsque cela arrive le disjoncteur saute.

### 2.2.4 Domaine de tension BTsup

En général, les bâtiments (écoles, piscines . . .) sont raccordés en Basse Tension  $> 36$  (BTsup). Les options possibles sont : Courte Utilisation (CU) et Longue Utilisation (LU); et les catégories temporelles sont : heure pleine saison haute (1), heure creuse saison haute (2), heure pleine saison basse (3) et heure creuse saison basse (4).

Les puissances souscrites  $x_i$  doivent prendre des valeurs supérieures à 36kVA. Afin d'assurer la stabilité de la charge du réseau, le TURPE5 impose que les puissances souscrites dans les catégories de forte consommation (par exemple les heures pleines saison haute) doivent être inférieures aux puissances souscrites dans les catégories de faible consommation (par exemple les heures creuses saison basse). Il en résulte une série de contraintes de la forme  $x_i \leq x_{i+1}$ , définissant un ordre total sur les valeurs  $x_i$ . La CS vaut  $\sum_{i=1}^4 s_i * x_i$ , pour des coefficients  $s_i$  en €/kVA donnés et  $x_i$  la puissance souscrite de la catégorie temporelle  $i$ . La CMDPS vaut chaque mois  $10.20 * h$ , sur la base de la durée de dépassement  $h$  (en heures).

### 2.2.5 Domaine de tension HTA

En général, les grands bâtiments (usines, ateliers, centres commerciaux . . .) sont raccordés en Haute Tension type A (HTA). Le PSCE du domaine de tension HTA est celui faisant intervenir le plus d'argent. Les options possibles sont : Courte Utilisation (CU) et Longue Utilisation (LU); et les  $K = 5$  catégories temporelles sont : pointe (1), heure pleine saison haute (2), heure creuse saison haute (3), heure pleine saison basse (4) et heure creuse saison basse (5). La pointe est une catégorie temporelle spéciale accessible uniquement par un raccordement HTA, et sont fixées localement de décembre à février inclus, à raison de 2 heures le matin et de 2 heures le soir, à l'exception des dimanches.

Soit  $T = 52560$  le nombre total de pas de temps de 10 minutes et  $T_i$  le nombre total de pas de temps dans la catégorie temporelle  $i$ . Soit  $M = \{1, \dots, 12\}$  l'ensemble de tous les mois de la période d'étude, et  $\forall i \in \{1, \dots, K\}$ ,  $M_i$  est le sous-ensemble de  $M$  correspondant à la catégorie  $i$ . Par exemple, pour la catégorie "pointe" (avec  $i = 1$ ),  $M_1 = \{12, 1, 2\}$  correspond aux trois mois décembre, janvier et février. Chaque pas de temps  $t$  appartient donc à une catégorie  $i$  et à un mois  $m \in M_i$ , noté  $t \in T_{i,m}$ , et pour chaque  $t$  il y a une donnée de consommation

électrique  $c_t$  (aussi dénotée par "puissance appelée").

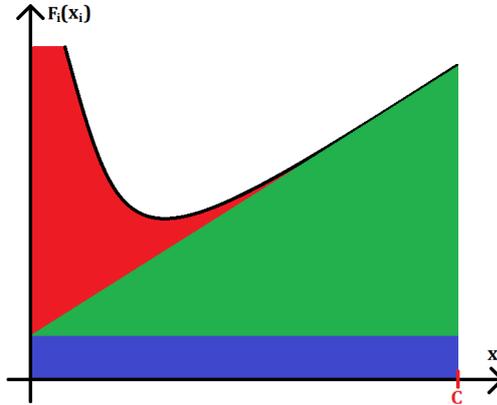
Le PSCE consiste à déterminer un vecteur d'entier non négatif à  $K$  dimensions  $x = (x_1, \dots, x_K)$ , minimisant la facture ( $F = CS + CMDPS$ ) due par le consommateur. Pour chaque catégorie  $i$ , la puissance souscrite correspondante représente des seuils de consommation ( $x_i \geq 0$  en kW), vus comme un engagement de consommation maximale pour chaque pas de temps de 10 minutes. Pour la stabilité du réseau, les valeurs  $x_i$  sont soumises à une contrainte d'ordre total (i.e.  $x_1 \leq x_2 \leq \dots \leq x_K$ ). En effet, la présence de cette contrainte pousse les consommateurs à choisir des puissances souscrites "similaires" entre les différentes catégories temporelles, et donc réduire théoriquement le gap de puissance appelée entre la saison haute et la saison basse.

La part fixe de la CS, appelée l'abonnement, est égal à  $\sum_i s_i * x_i$  pour des coefficient  $s_i > 0$  donnés (en €/kW/an), ce coût est donc une fonction linéaire croissante des puissances souscrites. Une demande excessive de consommation est autorisée, appelée dépassement de puissance souscrite, mais dans ce cas des coûts supplémentaires de pénalité sont dus. La quantité dépassée durant  $t$  est notée  $\delta_t(x_i) = \max(0, c_t - x_i)$ . Pour chaque catégorie  $i$  et chaque mois  $m \in M_i$ , le coût de pénalité est  $p_i \sqrt{\sum_{t \in T_{i,m}} \delta_t(x_i)^2}$  pour des  $p_i > 0$  donnés (en €/kWh), qui est une fonction décroissante de  $x_i$ . Tous les coûts, non liés à la puissance souscrite (par exemple la fourniture, le transport et les taxes), appelés "coût fixe" ne sont pas nécessaires pour l'analyse. La fonction objectif du problème est le coût total annuel  $F(x)$  défini comme la somme des coûts annuels  $F_i(x_i)$  pour les différentes catégories  $i \in \{1, \dots, K\}$  :

$$F_i(x_i) = s_i * x_i + p_i \sum_{m \in M_i} \sqrt{\sum_{t \in T_{i,m}} \delta_t(x_i)^2}. \quad (2.1)$$

Une représentation graphique de la fonction correspondant à [Équation \(2.1\)](#) est donnée dans la [Figure 2.1](#). L'abonnement CS est montré en vert, le dépassement CMDPS est montré en rouge et les coûts fixes, qu'on ignore dans l'analyse, sont montrés en bleu.

Soit  $\underline{c}_i$  (resp.  $\bar{c}_i$ ) la plus petite (resp. la plus grande) valeur de la puissance appelée pendant les pas de temps de la catégorie temporelle  $i$ . Nous désignons ensuite par  $\underline{c}$  (resp.  $\bar{c}$ ) la plus petite (resp. la plus grande) puissance appelée parmi toute l'instance. La nature du PSCE indique assez naturellement que les termes  $x_i$  de la solution seront entre les bornes  $\underline{c}$  et  $\bar{c}$ . On note  $C = \bar{c} - \underline{c} + 1$ . Ce paramètre

FIGURE 2.1 – Fonction objectif pour la catégorie  $i$  :  $F_i(x_i)$ 

$C$  et le nombre de catégories  $K$  seront utiles pour exprimer la complexité dans le pire des cas des algorithmes.

Pour un domaine de tension donné, la minimisation du coût la souscription de contrat d'électrique correspond à choisir une option tarifaire et ses puissances souscrites, ainsi il faut résoudre le PSCE pour chacune des options proposées.

Le PSCE peut alors être formulé comme le programme mathématique  $\mathcal{P}$  suivant :

$$\mathcal{P} : \min F(x) = \sum_{i=1}^K F_i(x_i) \quad (2.2)$$

$$s.t. \quad x_i \leq x_{i+1}, \quad 1 \leq i \leq K-1, \quad (2.3)$$

$$\underline{c} \leq x_i \leq \bar{c}, \quad 1 \leq i \leq K \quad (2.4)$$

$$x_i \in \mathbb{N}, \quad 1 \leq i \leq K \quad (2.5)$$

La fonction objectif [Équation \(2.2\)](#) est convexe et séparable comme l'indique la [Proposition 2.1](#). Les contraintes [Équation \(2.3\)](#) sont appelées *les contraintes d'ordre* et les contraintes [Équation \(2.4\)](#) sont appelées *les contraintes de borne*. Une solution  $\mathbf{x} = (\mathbf{x}_1, \dots, \mathbf{x}_K)^T$  pour  $\mathcal{P}$  est un vecteur d'entier non-nul de dimensions  $K$  satisfaisant les contraintes [Équation \(2.3\)](#) et [Équation \(2.4\)](#).

**Proposition 2.1.** *Chaque  $F_i$  est convexe par rapport à  $x_i$ .*

*Démonstration.* Pour tout  $t \in \{1, \dots, T\}$ , soit  $g_t(x_i)$  une fonction convexe univariée telle que  $g_t(x_t) \geq 0$  pour tout  $x_i \geq 0$ .

Pour tout  $x \in \mathbb{R}^T$ , soit  $G(x) = \sqrt{\sum_{t=1}^T g_t(x_t)^2}$  la norme euclidienne ( $L_2$ ) de

$$\begin{pmatrix} g_1(x_1) \\ \vdots \\ g_T(x_T) \end{pmatrix} \text{ i.e. } G(x) = \left\| \begin{pmatrix} g_1(x_1) \\ \vdots \\ g_T(x_T) \end{pmatrix} \right\|. \text{ Soit } \bar{x} = \begin{pmatrix} \bar{x}_1 \\ \vdots \\ \bar{x}_T \end{pmatrix} \geq 0, \hat{x} = \begin{pmatrix} \hat{x}_1 \\ \vdots \\ \hat{x}_T \end{pmatrix} \geq 0 \text{ et } \lambda \in [0, 1].c$$

D'après la convexité des fonctions  $g_t$  et le fait que la norme est une fonction non décroissante sur l'ensemble des vecteurs non négatifs :

$$\begin{cases} g_1(\lambda\bar{x}_1 + (1-\lambda)\hat{x}_1) \leq \lambda g_1(\bar{x}_1) + (1-\lambda)g_1(\hat{x}_1) \\ \vdots \\ g_T(\lambda\bar{x}_T + (1-\lambda)\hat{x}_T) \leq \lambda g_T(\bar{x}_T) + (1-\lambda)g_T(\hat{x}_T) \end{cases}$$

Ainsi :

$$\left\| \begin{pmatrix} g_1(\lambda\bar{x}_1 + (1-\lambda)\hat{x}_1) \\ \vdots \\ g_T(\lambda\bar{x}_T + (1-\lambda)\hat{x}_T) \end{pmatrix} \right\| \leq \left\| \begin{pmatrix} \lambda g_1(\bar{x}_1) + (1-\lambda)g_1(\hat{x}_1) \\ \vdots \\ \lambda g_T(\bar{x}_T) + (1-\lambda)g_T(\hat{x}_T) \end{pmatrix} \right\|$$

Maintenant, d'après la convexité de la norme euclidienne, il s'ensuit :

$$\left\| \begin{pmatrix} \lambda g_1(\bar{x}_1) + (1-\lambda)g_1(\hat{x}_1) \\ \vdots \\ \lambda g_T(\bar{x}_T) + (1-\lambda)g_T(\hat{x}_T) \end{pmatrix} \right\| \leq \lambda \left\| \begin{pmatrix} g_1(\bar{x}_1) \\ \vdots \\ g_T(\bar{x}_T) \end{pmatrix} \right\| + (1-\lambda) \left\| \begin{pmatrix} g_1(\hat{x}_1) \\ \vdots \\ g_T(\hat{x}_T) \end{pmatrix} \right\|$$

En conséquence, l'inégalité suivante est démontrée :

$$\left\| \begin{pmatrix} g_1(\lambda\bar{x}_1 + (1-\lambda)\hat{x}_1) \\ \vdots \\ g_T(\lambda\bar{x}_T + (1-\lambda)\hat{x}_T) \end{pmatrix} \right\| \leq \lambda \left\| \begin{pmatrix} g_1(\bar{x}_1) \\ \vdots \\ g_T(\bar{x}_T) \end{pmatrix} \right\| + (1-\lambda) \left\| \begin{pmatrix} g_1(\hat{x}_1) \\ \vdots \\ g_T(\hat{x}_T) \end{pmatrix} \right\|$$

et à partir de là, nous concluons que :

$$\sqrt{\sum_{t=1}^T g_t(\lambda\bar{x}_t + (1-\lambda)\hat{x}_t)^2} \leq \lambda \sqrt{\sum_{t=1}^T g_t(\bar{x}_t)^2} + (1-\lambda) \sqrt{\sum_{t=1}^T g_t(\hat{x}_t)^2},$$

ce qui prouve la propriété de convexité désirée de  $G$ .

Pour chaque catégorie  $i$ , le coût d'abonnement est linéaire en  $x_i$  et le coût de

pénalité est de la forme :  $p_i G_i(x_i)$  avec  $G_i(x_i) = \sqrt{\sum_{t=1}^{T_i} g_t(x_i)^2}$ , où  $g_t(x_i) = \delta_t(x_i)$  est une fonction convexe de  $x_i$ . Ainsi, pour chaque catégorie  $i$ ,  $F_i$  est convexe par rapport à  $x_i$ .  $\square$

## 2.3 Spécificité du problème : une grande quantité de données à traiter

Il est montré que le PCSE peut se ramener à un problème de minimisation de flot de coût convexe à objectif séparable ou à un problème convexe à objectif séparable sous des contraintes d'ordre total, mais la présence d'un très grande quantité de donnée à traiter ouvre la possibilité à de nouvelles approches.

### 2.3.1 Problème de flot de coût convexe à objectif séparable minimum

Le PSCE qu'on considère peut être reformulé comme un problème d'optimisation de flot de coût convexe à objectif séparable minimal dans un graphe comportant  $K$  nœuds et  $2K - 2$  arcs en utilisant la méthode suivante.

Le graphe à considérer contient  $K$  nœuds numérotés  $1, 2, \dots, K$  et un ensemble de  $2K - 1$  arcs, décomposés en  $K$  'arcs primaires' et  $K - 1$  'arcs secondaires'.

- L'ensemble des arcs primaires contient les  $K - 1$  arcs de la forme  $(i, i + 1)$  pour  $i = 1, \dots, K - 1$ , ainsi qu'un 'arc de retour'  $(K, 1)$ . La valeur du flot  $x_i$  sur chacun de ces arcs (y compris  $x_K$  pour l'arc de retour) doit respecter les contraintes de limite  $\underline{c} \leq x_i \leq \bar{c}$  (et  $\underline{c} \leq x_K \leq \bar{c}$ ) et la fonction de coût correspondante est  $F_i$  ( $F_K$  pour l'arc de retour);
- l'ensemble des arcs secondaires est composé de  $K - 1$  arcs de la forme  $(1, j)$  pour  $j = 2, \dots, K$ . La valeur du flot  $s_j$  sur chacun de ces arcs doit respecter les contraintes de limite  $\underline{c} \leq s_j \leq \bar{c}$ , et la fonction de coût associée est identiquement 0.

Il reste à déterminer une circulation de coût minimal sur le graphe ci-dessus. On observe que les valeurs des flots  $x_i$  sur les arcs primaires jouent le rôle de la valeur de la puissance souscrite, alors que les valeurs des flots  $s_j$  sur les arcs secondaires jouent le rôle de variables d'écart pour les contraintes d'ordre  $x_{j-1} \leq x_j$  pour  $j = 2, \dots, K$ .

Ainsi, afin de résoudre  $\mathcal{P}$ , on peut envisager d'appliquer certaines des procédures de résolution existantes, en particulier :

- l'algorithme proposé dans [77, 78] qui, dans notre cas, conduirait à une complexité dans le pire des cas de  $O(K^3 \log C)$  ;
- les algorithmes proposés dans [2] et dans [65] qui conduiraient à une complexité dans le pire des cas de  $O(K^2 \log(K) \log(KC))$ .

Les algorithmes de [2] et [65] ont besoin de structures de données avancées telles que des arbres dynamiques pour atteindre la complexité dans le pire des cas de  $O(K^2 \log(K) \log(KC))$ . Sans elles, la complexité dans le pire des cas resterait de  $O(K^3 \log C)$ . Pour une étude des algorithmes de résolution des problèmes d'optimisation de réseaux de coûts convexes et des problèmes d'optimisation non linéaires connexes, nous renvoyons vers [53].

### 2.3.2 Problème de régression isotonique

Étant donné  $K$  données  $c_1, \dots, c_K$ , le problème de régression isotonique (PRI ou *Isotonic Regression Problem*) consiste à trouver  $K$  valeurs  $x_1, \dots, x_K$  minimisant le carré de la distance entre la solution et les données sous les contraintes d'ordre  $x_1 \leq \dots \leq x_K$ . C'est un problème bien connu dans les statistiques, dans la recherche opérationnelle et dans le traitement d'images, il est formulé comme suit :

$$\begin{aligned} \min \quad & \sum_i (x_i - c_i)^2 \\ \text{s.t.} \quad & x_1 \leq x_2 \leq \dots \leq x_K. \end{aligned}$$

Un algorithme itératif simple, le *pool adjacent violators* (PAV), introduit pour la première fois par [9], peut résoudre facilement ce problème. Différentes versions de régression isotonique sont étudiées en utilisant des fonctions objectif différentes, utilisant par exemple les normes  $L_0$ ,  $L_1$ ,  $L_2$  ou  $L_\infty$  [97]. Ils peuvent être résolus par le *minimum lower sets* [25], le *minimum violator algorithm* [63]. De même, le problème peut être résolu avec un algorithme d'identification d'ensemble de contrainte actif [17]. D'autres travaux ont été effectués autour du PRI, on peut noter : le problème de sac à dos de régression isotonique [16], la régression médiane isotonique [28, 84], ou encore la méthode de régression monotonique réduite [92].

Le problème de régression isotonique généralisé est obtenu lorsque la fonction objectif est remplacée par une fonction convexe séparable quelconque [3].

### 2.3.3 Problème convexe à objectif séparable sous des contraintes d'ordre total

Le PSCE appartient également à la classe des problèmes de minimisation de fonctions convexes à objectif séparables sous des contraintes d'ordre total. Cette classe de problèmes apparaît dans le contexte de la régression isotonique, l'un des algorithmes les plus efficaces pour résoudre le PRI généralisé a été proposé par Ahuja, R. K., et Orlin, J. B. (2001) [3] sous le nom de 'scaling\_PAV'. Dans scaling\_PAV, on considère un ordre total entre des intervalles comme étant un équivalent de l'ordre total entre les variables. L'algorithme cherche à trouver  $K$  intervalles d'une certaine taille qui correspondent aux  $K$  variables de la solution optimale. Les intervalles sont tous initialement fixés à  $[\underline{c}, \bar{c}]$  et sont réduits de moitié après chaque itération. Il est possible que la mise à jour de deux des intervalles consécutives crée une violation de la contrainte d'ordre correspondante. Dans le cas échéant, l'algorithme annule les mises à jour des deux intervalles et va créer un unique intervalle qui remplace les deux intervalles précédents, ainsi les deux variables correspondantes seront forcées à être égales jusqu'à la fin de l'exécution de l'algorithme. Après  $\log_2(\frac{C}{\epsilon})$  itérations, tous les intervalles ont une longueur réduite inférieure à  $\epsilon$ , et une solution  $\epsilon$ -optimale est trouvée. En supposant, que l'évaluation de chacun des termes Équation (2.1) de l'objectif séparable peut être calculée en temps constant  $O(1)$ , la complexité dans le pire des cas de scaling\_PAV est de  $O(K \log(C))$ .

### 2.3.4 Le PCSE face au Big Data

De toute évidence, l'algorithme scaling\_PAV pourrait être appliqué au PSCE traité ici. Cependant, dans ce contexte d'application, l'évaluation de la complexité résultante doit être complètement reconsidérée, car un examen attentif révèle que l'effort de calcul nécessaire pour calculer chaque valeur  $F_i(x_i)$  impliquée dans la fonction objectif Équation (2.2) ne peut pas supposé être  $O(1)$ . En effet, de nos jours les applications font face à un très grand volume de donnée, c'est pourquoi il est fortement recommandé de réfléchir sur la stratégie de l'évaluation des fonctions

objectif.

À partir de l'expression donnée dans l'Équation (2.1), il est facile de voir que le calcul de  $F_i(x_i)$  pour toute valeur donnée de  $x_i$ , nécessite  $O(N^+(x_i))$  opérations arithmétiques, où  $N^+(x_i)$  est défini comme le nombre de pas de temps  $t$  dans  $T_i$  tel que  $c_t > x_i$ . Ainsi, pour de grandes valeurs de  $x_i$  (i.e. des valeurs proches de  $\max(\{c_t, t \in T_i\})$ ,  $N^+(x_i)$  et le temps de calcul sont largement réduits. En revanche, pour des valeurs plus petites de  $x_i$ , le  $N^+(x_i)$ , e.g. du même ordre de grandeur que  $|T_i|$  (le nombre total de pas de temps dans la définition de  $F_i$ ), conduit ainsi à un temps de calcul probablement beaucoup plus important. Pour une discussion plus approfondie et une illustration de cette dépendance de  $N^+(x_i)$  par rapport à  $x_i$ , veuillez vous référer à la Sous-section 3.1.2. Clairement, puisque  $|T_i|$  peut être beaucoup plus grand que  $K$ , i.e. le nombre de variables de décision, ce paramètre s'avère être un facteur clé à prendre en compte dans l'évaluation d'un algorithme de résolution du PSCE. Ainsi, dans la recherche de l'efficacité computationnelle dans la résolution du PSCE, la dépendance de  $N^+(x_i)$  (une mesure de l'effort de calcul) par rapport à  $x_i$  doit être prise en compte.

Dans le chapitre suivant, nous proposons un algorithme spécifique pour résoudre le PSCE qui s'avérera être plus efficace que `scaling_PAV`, grâce à une exploitation appropriée des caractéristiques spécifiques du problème soulignées ci-dessus. L'algorithme proposé, que nous appellerons `OPTIM_SP`, est basé sur le même schéma que `scaling_PAV` i.e. la détection des ensembles de contraintes actives de manière itérative. Cependant, sa mise en œuvre fait un usage essentiel d'une procédure de recherche dichotomique asymétrique étendue, grâce à laquelle l'effort de calcul requis à chaque itération pour minimiser la partie pertinente de la fonction objectif Équation (2.2) peut être considérablement réduit, par rapport à la recherche dichotomique classique.

Il convient également de mentionner ici que la possibilité de recourir à cette procédure de recherche binaire étendue n'est pas applicable dans l'algorithme `scaling_PAV` pour la raison suivante : lors de l'exécution de `scaling_PAV`, à tout moment, tous les intervalles doivent avoir la même longueur afin de pouvoir procéder à un remplacement des intervalles violant une contrainte d'ordre. Ce qui ne peut être le cas en utilisant la recherche dichotomique asymétrique étendue.

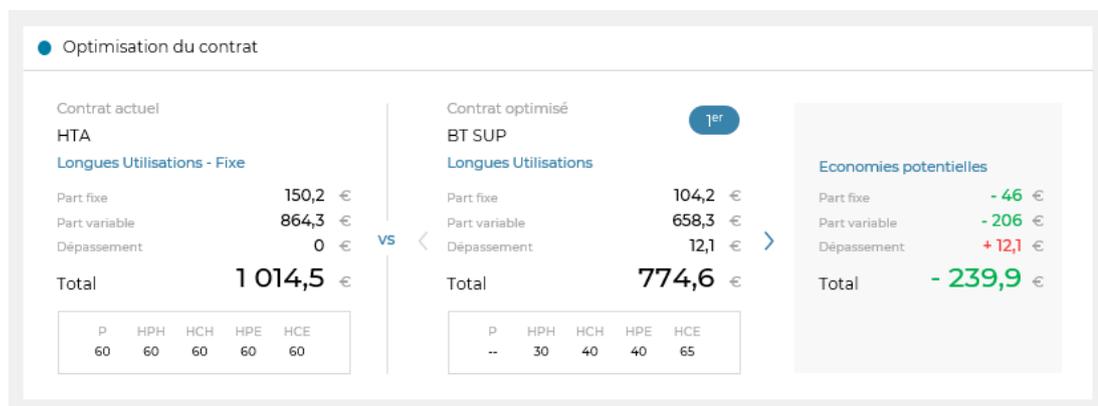


FIGURE 2.2 – Widget de l'optimisation de la puissance souscrite monosite

## 2.4 Industrialisation du PSCE

L'optimisation de la puissance souscrite est industrialisée et proposée sur le portail d'Énergisme pour ses clients afin de proposer des économies de facture d'électricité. Lorsqu'une requête d'optimisation de la puissance souscrite est initiée par un client, le système va récupérer les données du contrat actuel et les données de consommation avec l'API mise en place par ENEDIS ou dans les bases de données d'Energisme. Ensuite, un prétraitement va vérifier que l'instance respecte les conditions minimales de l'optimisation (quantité et qualité de la donnée etc...) et va aussi nettoyer la donnée pour correspondre au format attendu. Finalement, pour chacun des types de raccordement possibles (BTinf, BTsup et HTA) et pour chacune des options tarifaires possibles (Courte Utilisation, Moyenne Utilisation, Longue Utilisation etc...), le logiciel procède à l'optimisation de la puissance souscrite en question. Pour les raccordements en BTinf, le problème se ramène à la recherche de la valeur de la plus grande consommation. Pour les raccordements en BTsup, le problème se ramène à la recherche d'un plus court chemin dans un graphe où les sommets représentent les valeurs possibles des puissances souscrites. Pour les raccordements en HTA, le problème se ramène à la résolution d'un problème convexe à objectif séparable et utilisant l'algorithme qu'on propose dans le [Chapitre 3](#).

Un visuel du widget de l'optimisation du portail est montré dans la [Figure 2.2](#). Énergisme propose aussi une version multi-PDL (dite *multisite*) de l'optimisation de la puissance souscrite (jusqu'à 70000 PDLs pour certain client). Un visuel du widget de l'optimisation multisite du portail est montré dans la [Figure 2.3](#). Le

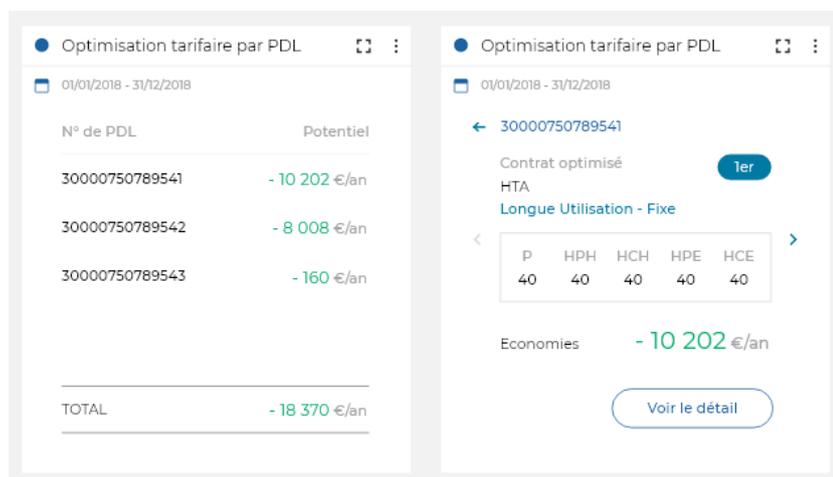


FIGURE 2.3 – Widget de l’optimisation de la puissance souscrite multisite



FIGURE 2.4 – Widget du pareto des économies potentielles par site

widget de la [Figure 2.4](#) est mise à disposition afin d'afficher les PDLs apportant les plus grosses économies par site.

## 2.5 Résumé du chapitre

Nous présentons le problème de sélection des contrats d'électricité des points de livraison raccordés en Haute Tension type A au sens du TURPE d'ENEDIS. Le problème consiste à trouver les puissances souscrites d'un contrat d'électricité pour un client en fonction de son historique de consommation d'électricité sur une période de temps fixe. L'objectif est d'optimiser la facture d'électricité composée du coût de la fourniture, du coût de la souscription du contrat d'électricité et des pénalités dues aux dépassements de puissance souscrite et de certain coût fixe.

Le PSCE peut être formulé comme un problème d'optimisation convexe à objectif séparable soumis à des contraintes d'ordre total. En raison de cette structure spéciale, le PSCE est un cas particulier de deux classes de problèmes d'optimisation convexe à objectif séparable bien connues, à savoir le flot de coût minimum avec un coût convexe à objectif séparable et la minimisation de fonctions convexes à objectif séparables sous des contraintes de chaîne. Ces deux classes sont bien traitées dans la littérature et peuvent être résolues en temps polynomial. [[2](#), [3](#), [18](#), [65](#), [77](#), [78](#)].



# Chapitre 3

## Algorithme pour le problème de souscription de contrat d'électricité

### Sommaire

---

3.1	OPTIM_SP : un algorithme ajustable . . . . .	<b>36</b>
3.1.1	Présentation générale de l'algorithme . . . . .	37
3.1.2	Recherche par dichotomie ajustable . . . . .	37
3.1.3	Résoudre $\mathcal{P}^=(\emptyset)$ . . . . .	41
3.1.4	Résoudre $\mathcal{P}^=(B)$ avec $B \neq \emptyset$ . . . . .	41
3.1.5	Validité de OPTIM_SP . . . . .	41
3.1.6	Complexité de OPTIM_SP . . . . .	42
3.2	Résultats expérimentaux . . . . .	<b>43</b>
3.2.1	Jeux de données et solutions optimales . . . . .	43
3.2.2	Calcul de complexité de OPTIM_SP et scaling_PAV . . . . .	46
3.2.3	Évaluation comparative de OPTIM_SP et scaling_PAV . . . . .	46
3.3	Résumé du chapitre . . . . .	<b>49</b>

---

Nous proposons un nouvel algorithme pour le PSCE qui est spécialement conçu pour réduire l'effort de calcul sur des données historiques à grande échelle, car en effet le temps nécessaire à l'évaluation de la fonction objectif ne peut plus être supposé être  $O(1)$ . Nous présentons des résultats numériques montrant que notre algorithme surpasse celui de Ahuja, R. K., et Orlin, J. B. (2001) [3], l'un des meilleures algorithmes existants pour cette classe de problème, lorsqu'il est appliqué aux données de consommation de divers types de clients pour le PSCE.

### 3.1 OPTIM\_SP : un algorithme ajustable basé sur une approche d'ensemble de contraintes actives

Pour  $k \in \{1, \dots, K-1\}$ , la  $k^{\text{eme}}$  contrainte d'ordre désigne la contrainte  $x_k \leq x_{k+1}$  parmi les contraintes d'ordre total. Pour  $k \in \{0, \dots, K-1\}$ , soit  $\mathcal{P}^k$  désignant la relaxation du problème  $\mathcal{P}$  où les  $K - 1 - k$  dernières contraintes d'ordre sont relaxées, i.e. que les contraintes d'ordre dans  $\mathcal{P}^k$  sont seulement les  $k$  premières contraintes d'ordre :

$$\begin{aligned} \mathcal{P}^k : \quad & \min F(x) \\ \text{s.t.} \quad & x_i \leq x_{i+1}, & 1 \leq i \leq k, \\ & \underline{c} \leq x_i \leq \bar{c}, & 1 \leq i \leq K \\ & x_i \in \mathbb{N}, & 1 \leq i \leq K \end{aligned}$$

Notons que  $\mathcal{P}^0$  est la relaxation de  $\mathcal{P}$  sans les contraintes d'ordre et que  $\mathcal{P}^{K-1}$  est  $\mathcal{P}$  lui-même.

Plus généralement, pour  $B \subseteq \{1, \dots, K - 1\}$ , notons  $\mathcal{P}(B)$  la relaxation de  $\mathcal{P}$  où toutes les contraintes d'ordre sont relaxées sauf celles indexées dans  $B$  :

$$\begin{aligned} \mathcal{P}(B) : \quad & \min F(x) \\ \text{s.t.} \quad & x_i \leq x_{i+1}, & i \in B, \\ & \underline{c} \leq x_i \leq \bar{c}, & 1 \leq i \leq K \\ & x_i \in \mathbb{N}, & 1 \leq i \leq K \end{aligned}$$

En outre, nous désignons par  $\mathcal{P}^=(B)$  la restriction de  $\mathcal{P}(B)$  où les contraintes d'ordre dans  $B$  sont fixées à l'égalité :

$$\begin{aligned} \mathcal{P}^=(B) : \quad & \min F(x) \\ \text{s.t.} \quad & x_i = x_{i+1}, & i \in B, \\ & \underline{c} \leq x_i \leq \bar{c}, & 1 \leq i \leq K \\ & x_i \in \mathbb{N}, & 1 \leq i \leq K \end{aligned}$$

Notons que si  $B = \emptyset$ , alors  $\mathcal{P}^=(B) = \mathcal{P}^0$ .

### 3.1.1 Présentation générale de l'algorithme

Cette petite sous-section a pour but de montrer une vue d'ensemble de l'algorithme proposé ainsi que ses notations.

Dans la suite,  $x$  désigne une variable dans la fonction objectif ou les contraintes alors que  $\mathbf{x}$  désigne une solution composé de variables entières durant l'exécution de l'algorithme. La différentiation permet de mieux suivre le déroulement des algorithmes.

Notre algorithme OPTIM\_SP résout initialement  $\mathcal{P}^0$  puis effectue  $K-1$  itérations résolvant ainsi successivement  $\mathcal{P}^1, \mathcal{P}^2, \dots, \mathcal{P}^{K-1}$ . Pour  $k = 0, \dots, K-1$ , soit  $\mathbf{x}^k = (\mathbf{x}_1^k, \dots, \mathbf{x}_K^k)^T$  la solution optimale de  $\mathcal{P}^k$  trouvée par OPTIM\_SP, et soit  $B^k$  l'ensemble des indices des contraintes d'ordre actives dans  $\mathcal{P}^k$  associées à  $\mathbf{x}^k$ , i.e. les contraintes d'ordre dans  $\mathcal{P}^k$  que  $\mathbf{x}^k$  satisfait à l'égalité.

OPTIM\_SP peut être formellement énoncé comme l'Algorithme 3.1. À l'initialisation, il n'y a aucune contrainte d'ordre dans le problème, ainsi la solution  $\mathbf{x} \leftarrow \mathbf{x}^0$  est bien la solution optimale de  $\mathcal{P}^0$ . Ensuite, à chaque itération  $k = 1, \dots, K-1$ , on cherche la contrainte d'ordre  $x_i \leq x_{i+1}$  du plus grand indice  $i$  dans  $\mathcal{P}^k$  violée par la solution courante  $\mathbf{x}$ , puis on met à jour  $B \leftarrow B \cup \{i\}$ , et on trouve un nouveau  $\mathbf{x}$  en résolvant  $\mathcal{P}^=(B)$  (respectant donc la contrainte précédemment violée). Si aucun indice ne peut être trouvé, alors la solution courant est solution de  $\mathcal{P}^k$ , donc on termine l'itération  $k$  en posant  $\mathbf{x}^k \leftarrow \mathbf{x}$  et  $B^k \leftarrow B$  et on .

Notons que minimiser  $F(x)$  par rapport à  $B$  signifie minimiser  $F(x)$  sous des contraintes d'ordre dans  $B$  fixées à l'égalité et des contraintes de limite. Par exemple, si  $B = 1$ , alors il faut chercher le minimum de  $F_1(x_1) + F_2(x_2)$  tel que  $x^* = x_1 = x_2$  et  $\underline{c} \leq x^* \leq \bar{c}$ .

### 3.1.2 Recherche par dichotomie ajustable pour l'optimisation de fonctions convexes univariées

Chaque étape de l'Algorithme 3.1 requiert la résolution du problème  $\mathcal{P}^=(B)$ , soit la résolution de plusieurs problèmes de minimisation de fonction convexe univariée entière sur un intervalle donné, une résolution pour chaque index n'étant pas dans  $B$  et pour chaque séries consécutives d'index étant dans  $B$  du le problème.

---

**Algorithme 3.1** Pseudo code pour OPTIM\_SP

---

```

1:  $B \leftarrow \emptyset$ .
2: Résoudre  $\mathcal{P}^=(B)$  et fixer  $\mathbf{x}$  égal à la solution optimale obtenue.
3: pour  $k$  de 1 à  $K-1$  faire
4:    $b \leftarrow True$ .
5:   répéter
6:     Trouver le plus grand indice  $1 \leq i \leq k$  tel que  $\mathbf{x}_i > \mathbf{x}_{i+1}$ .
7:     si un tel indice existe alors
8:        $B \leftarrow B \cup \{i\}$ 
9:       Résoudre  $\mathcal{P}^=(B)$  et mettre à jour  $\mathbf{x}$ .
10:    sinon
11:       $b \leftarrow False$ .
12:    fin si
13:  jusqu'à  $b = False$ 
14: fin pour
15: retourner  $\mathbf{x}$ 

```

---

Nous proposons ainsi une procédure de recherche par dichotomie étendue appelée A\_SEARCH, formellement formulée comme l'Algorithme 3.2. A\_SEARCH est une version ajustable de la recherche par dichotomie qui dépend du paramètre d'entrée  $a > 0$ . Ainsi, pour un intervalle  $[y, \bar{y}]$  et une fonction convexe univariée donnée  $F_I$  où  $I \subset \{1, \dots, K\}$  tel que  $F_I(y) = \sum_{j \in I} F_j(y)$ , la procédure A\_SEARCH renvoie :

$$y_{opt} = \underset{\substack{y \leq y \leq \bar{y} \\ y \text{ entier}}}{\operatorname{argmin}} F_I(y).$$

La recherche par dichotomie ajustable commence par une borne inférieure  $y$  et une borne supérieure  $\bar{y}$ . Jusqu'à ce que  $y_{opt}$  soit trouvé, l'algorithme calcule le point intermédiaire entier  $ymid = \lfloor (y+a * \bar{y}) / (1+a) \rfloor$  selon le paramètre  $a$ , et calcule les valeurs  $left\_cost$  et  $right\_cost$  pour les deux voisins à  $ymid-\epsilon$  et  $ymid+\epsilon$ , puis il réduit l'intervalle de recherche en mettant à jour soit  $y$  soit  $\bar{y}$ . Finalement, la recherche par dichotomie ajustable fonctionne de manière similaire à la version classique entière, la différence est uniquement dans le choix du point intermédiaire entier de chaque itération.

Notons que la procédure de recherche par dichotomie standard correspond à un choix de  $a = 1$  dans A\_SEARCH. Pour cette valeur de  $a$ ,  $ymid$  est le milieu de l'intervalle  $[y, \bar{y}]$ . Pour des  $a > 1$ ,  $ymid$  n'est plus le milieu de l'intervalle, et nous appelons ce cas la "recherche par dichotomie asymétrique".

La motivation derrière l'utilisation de  $a > 1$  provient de l'observation que

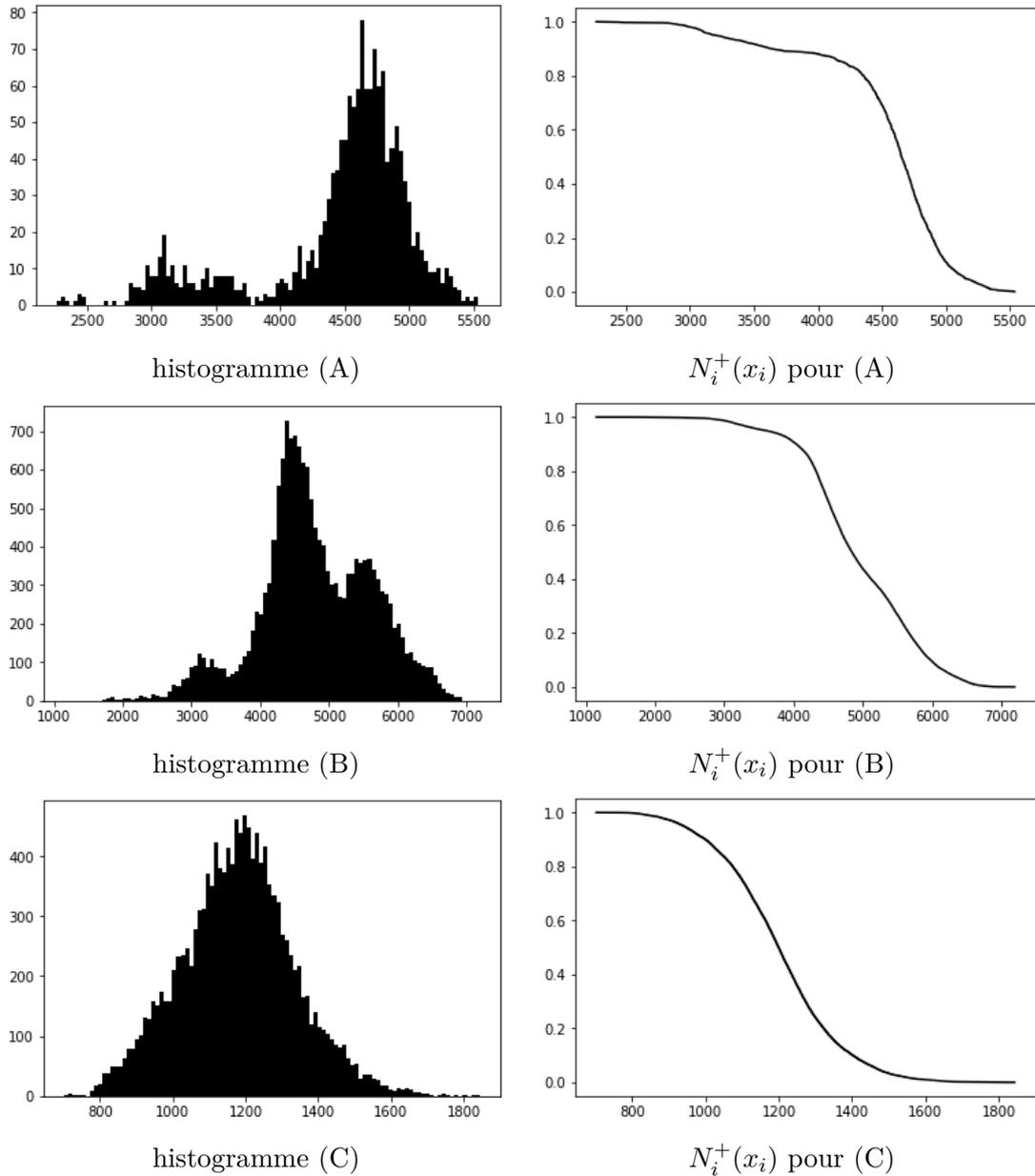


FIGURE 3.1 – Histogrammes et courbes montrant la dépendance de  $N_i^+(x_i)$  par rapport à  $x_i$  pour trois exemples qui font partie des ensembles de données D1 et D5 représentant les puissances appelées décrits dans la [Section 3.2](#)

---

**Algorithme 3.2** Pseudo code pour la recherche par dichotomie ajustable

---

```

1: fonction  $y_{opt} \leftarrow A\_SEARCH(F_I(y), \underline{y}, \bar{y}, a)$ 
2:    $\epsilon \leftarrow 0.1$ 
3:   répéter
4:      $ymid \leftarrow \lfloor (\underline{y} + a * \bar{y}) / (1 + a) \rfloor$ 
5:      $left\_cost \leftarrow F_I(ymid - \epsilon)$ 
6:      $right\_cost \leftarrow F_I(ymid + \epsilon)$ 
7:     si  $left\_cost \leq right\_cost$  alors
8:        $\bar{y} \leftarrow ymid$ 
9:        $a \leftarrow 1$ 
10:    sinon
11:       $\underline{y} \leftarrow ymid$ 
12:    fin si
13:  jusqu'à  $\underline{y} + 1 = \bar{y}$ 
14:   $y_{opt} \leftarrow ymid$ 
15:  retourner  $y_{opt}$ 
16: fin fonction

```

---

l'effort de calcul  $O(N_i^+(x_i))$  nécessaire pour calculer la valeur d'une composante de la fonction objectif telle qu'il est défini dans l'Équation (2.1) et dépend ainsi fortement de la valeur du paramètre  $x_i$ . Cette dépendance est illustrée dans la Figure 3.1 sur trois exemples correspondant aux ensembles de données  $D_1$  et  $D_5$ , deux des ensembles de données utilisés dans les expériences de calcul discutées dans la Section 3.2 d'application ci-dessous. Plus précisément, la Figure 3.1(A) correspond à l'ensemble de données  $D_1$  et  $i = 1$ ; la Figure 3.1(B) correspond à l'ensemble de données  $D_1$  et  $i = 5$ ; la Figure 3.1(C) correspond à l'ensemble de données  $D_5$  et  $i = 4$ .

Notons que la formulation du TURPE et les résultats empiriques de la Sous-section 3.2.1, nous avons un a priori sur  $y_{opt}$  qui devrait être une assez grande valeur, ce qui guide un peu la recherche. Ainsi, avec  $a > 1$  on applique des itérations de recherche par dichotomie asymétrique convergeant plus rapidement vers les grandes valeurs; puis si durant une itération la recherche par dichotomie ne continue pas sur (ne sélectionne pas) le plus petit sous-intervalle (i.e. celui à droite de  $ymid$ ), alors nous avons atteint la limite de notre a priori et nous continuons avec des itérations de recherche pas dichotomie classique en fixant  $a = 1$  (cf.ligne 9 de l'Algorithme 3.2). On peut vérifier que le nombre d'itérations dans  $A\_SEARCH$  est au plus  $\log_2(\bar{y} - \underline{y}) + 1$ . Dans la Section 3.2, la résolution du problème PSCE sera expérimenté avec des valeurs initiales de  $a$  entre 1 et 8.

### 3.1.3 Résoudre $\mathcal{P}^=(\emptyset)$

Pour réaliser la ligne 2, de l'Algorithme 3.1 de OPTIM\_SP, qui consiste à résoudre le problème  $\mathcal{P}^=(\emptyset)$ , i.e. résoudre  $K$  problèmes de minimisation de fonction convexe univariée entière. Nous appelons la fonction A\_SEARCH pour minimiser chaque  $F_i(x)$  pour  $i = 1, \dots, K$ , tel que  $\mathbf{x}_i = \text{A\_SEARCH}(F_i(x), \underline{c}_i, \bar{c}_i, a)$ . Puisque les fonctions sont convexes, on peut un peu réduire l'espace de recherche, en utilisant les intervalles  $[\underline{c}_i, \bar{c}_i]$  de  $i$  et non pas l'intervalle commun  $[\underline{c}, \bar{c}]$  du problème  $\mathcal{P}$  (rappel  $\underline{c}, \underline{c}_i, \bar{c}_i, \bar{c}$ ). Il est facile de voir qu'après exécution de la ligne 2,  $\mathbf{x}$  est une solution optimale de  $\mathcal{P}^=(B)$  avec  $B = \emptyset$ , car chacune des variables est le minimum d'un des termes de l'objectif convexe entier sans contrainte.

### 3.1.4 Résoudre $\mathcal{P}^=(B)$ avec $B \neq \emptyset$

La procédure correspondant à la ligne 9, de l'Algorithme 3.1 de OPTIM\_SP, consiste à résoudre  $\mathcal{P}^=(B)$  avec  $B \neq \emptyset$ . Supposons que nous soyons à l'itération  $k$  ( $1 \leq k \leq K - 1$ ) et que nous soyons prêts à exécuter la ligne 9 après avoir ajouté un indice  $i$  à  $B$  à la ligne 8. Soit  $B' = B \setminus \{i\}$ , i.e. la valeur de  $B$  avant l'exécution de la ligne 8. Soit  $\mathbf{x}'$  la solution optimale de  $\mathcal{P}^=(B')$ , i.e. la valeur de  $\mathbf{x}$  avant la ligne 9, on a alors  $\mathbf{x}'_i > \mathbf{x}'_{i+1}$ . Soit  $\underline{i}$  et  $\bar{i}$  respectivement les plus petits et les plus grands indices du bloc d'indices consécutifs dans  $B$  qui contient  $i$ . Soit  $I = \{\underline{i}, \underline{i} + 1, \dots, \bar{i}, \bar{i} + 1\}$ . La ligne 9 est exécutée en appelant simplement  $\text{A\_SEARCH}(F_I(x), \mathbf{x}'_{\underline{i}}, \mathbf{x}'_{\bar{i}}, a)$  où  $F_I(x) = \sum_{j \in I} F_j(x)$ . Cet appel de fonction renvoie une valeur entière  $y_{opt}$ , et la mise à jour de  $\mathbf{x}$  (à la ligne 9 de l'Algorithme 3.1) est effectuée comme suit :  $\mathbf{x}_j \leftarrow \mathbf{x}'_j$  pour  $j \notin I$  et  $\mathbf{x}_j \leftarrow y_{opt}$  pour  $j \in I$ .

Pour chaque index  $j \notin B$ , la valeur  $\mathbf{x}_j$  était déjà le minimum de  $F_j$ ; pour chaque série d'index consécutif dans  $J \subset B \cup I$ , la valeur  $\mathbf{x}_J$  était déjà le minimum de  $F_J$ ; pour  $I$ , la valeur  $y_{opt}$  est le minimum de  $F_I$ ; finalement  $\mathbf{x}$  est bien optimale pour  $\mathcal{P}^=(B)$ .

### 3.1.5 Validité de OPTIM\_SP

Une partition  $J = \{J_1, \dots, J_k\}$  de l'ensemble d'indices  $\{1, 2, \dots, K\}$  des variables du problème, est appelée *une partition par blocs* si ses éléments sont des sous-ensembles (ou blocs) d'indices consécutifs. Étant donné une solution  $\mathbf{x} \in \mathbb{R}^K$ ,  $\mathbf{x}$  est *conforme* à  $J$  si  $\mathbf{x}_i = \mathbf{x}_j$  pour tous  $i$  et  $j$  appartenant au même bloc dans  $J$ . La validité de OPTIM\_SP est basée sur des résultats bien connus dans la

littérature. En particulier, nous citons ici le [Lemme 3.1](#) suivant (lemme 2 dans [\[3\]](#) reformulé de manière appropriée pour s'adapter à notre contexte).

**Lemme 3.1.** *Si  $J = \{J_1, \dots, J_k\}$  est une partition en blocs de  $\{1, 2, \dots, K\}$  telle qu'il existe une solution  $\mathbf{x} \in \mathbb{R}^K$  conforme à  $J$  qui est également réalisable pour  $\mathcal{P}$  alors  $\mathbf{x}$  est une solution optimale pour  $\mathcal{P}$  [\[3\]](#).*

Finalement, nous pouvons affirmer le [Théorème 3.2](#) suivant :

**Théorème 3.2.** *La solution  $\mathbf{x}$  produite par OPTIM\_SP est optimale pour  $\mathcal{P}$ .*

*Démonstration.* Comme nous ne pouvons pas trouver un plus grand indice  $1 \leq i \leq K-1$  tel que  $\mathbf{x}_i > \mathbf{x}_{i+1}$ ,  $\mathbf{x}$  est réalisable pour  $\mathcal{P}$ . Soit  $\bar{B}$  la dernière valeur de  $B$  après la terminaison de OPTIM\_SP alors  $\bar{B}$  est l'ensemble des contraintes actives que  $\mathbf{x}$  satisfait à l'égalité. Construisons une partition de bloc  $J$  de  $\{1, 2, \dots, K\}$  à partir de  $\bar{B}$  comme suit :

- Pour chaque bloc d'indices consécutifs  $I$  dans  $\bar{B}$  avec  $\underline{i}$  et  $\bar{i}$  comme respectivement le plus petit et le plus grand indice dans  $I$ , désignons  $\{\underline{i}, \underline{i} + 1, \dots, \bar{i}, \bar{i} + 1\}$  comme un bloc dans  $J$ .
- Pour un indice  $1 \leq j \leq K$ , si ni  $j$  ni  $j - 1$  n'appartiennent à  $\bar{B}$ , alors désignons  $\{j\}$  comme un bloc élémentaire dans  $J$ .

Ainsi  $\mathbf{x}$  est une solution réalisable de  $\mathcal{P}$  conforme à  $J$ . Par conséquent, par le [Lemme 3.1](#),  $\mathbf{x}$  est optimale pour  $\mathcal{P}$ . □

### 3.1.6 Complexité de OPTIM\_SP

**Proposition 3.3.** *La complexité dans le pire des cas de OPTIM\_SP est de  $O(K^2 \log(C))$  à condition que l'évaluation de chaque  $F_i(x)$  à un point spécifique  $x$  puisse être effectuée en  $O(1)$ .*

*Démonstration.* L'exécution de OPTIM\_SP dans le pire des cas se produit lorsque l'ensemble  $B$  de la ligne 9 de OPTIM\_SP contient successivement  $1, 2, \dots, K-1$  éléments et il appelle  $K-1$  fois  $A\_SEARCH(F_I(x), x'_i, x'_i, a)$  à la ligne 9 avec les ensembles  $I$  (tels que définis dans la [Sous-section 3.1.4](#)) contenant successivement  $2, 3, \dots, K$  éléments. Ainsi, le nombre total de fois où les fonctions  $F_i(x)$  ( $i = 1, \dots, k$ ) sont minimisées dans les exécutions de  $A\_SEARCH$  est de  $\frac{K(K+1)}{2} - 1$ .

Nous devons également inclure la minimisation de chaque  $F_i(x)$  dans les  $K$  appels de `A_SEARCH` pour résoudre  $\mathcal{P}^0$  à la ligne 2 de `OPTIM_SP`. Par conséquent, il y a globalement  $\frac{K(K+1)}{2} - 1 + K = \frac{K(K+3)}{2} - 1$  évaluations de  $F_i(x)$  dans les appels de `A_SEARCH`.

Dans chaque minimisation de  $F_i(x)$  par `A_SEARCH`,  $F_i(x)$  est évalué au plus  $\log_2(C) + 1$  fois comme indiqué dans la section [Sous-section 3.1.2](#). Par conséquent, le nombre d'évaluations des fonctions  $F_i(x)$  est de  $O(K^2 \log(C))$  dans le pire des cas.  $\square$

D'après la [Proposition 3.3](#), la complexité dans le pire des cas de `OPTIM_SP` est de  $O(K^2 \log(C))$  ce qui est a priori moins intéressant que  $O(K \log(C))$  de `scaling_PAV`. Dans la prochaine section, nous montrons que dans la pratique `OPTIM_SP` est deux à trois fois plus rapide que `scaling_PAV` pour résoudre le PSCE.

## 3.2 Résultats expérimentaux

Cette section est consacrée à une étude expérimentale détaillée de l'algorithme `OPTIM_SP` sur une série d'instances réelles typiques du PSCE. Pour chaque instance, l'évolution du temps CPU est analysée en fonction de la valeur choisie pour le paramètre  $a$  introduit dans la [Sous-section 3.1.2](#). Les résultats obtenus sont également comparés à ceux qui seraient obtenus en utilisant l'algorithme `scaling_PAV` sur les mêmes données.

### 3.2.1 Jeux de données et solutions optimales

Dans nos expériences de calcul, nous considérons 5 jeux de données appelés  $D_1$  à  $D_5$ . Chaque jeu de données est défini en spécifiant les 52560 valeurs de la consommation d'électricité d'un utilisateur, sur des pas de temps de 10 minutes durant une année complète (2017). Cet ensemble de valeurs est décomposé en 5 catégories temporelles  $T_1$  à  $T_5$ . Chaque  $T_i$  est à son tour décomposé en plusieurs sous-ensembles, notés  $T_{i,m}$  pour un certain  $m$  dans  $M_i$ , correspondant aux mois pertinents durant lesquels la consommation a été observée. Les fonctions objectifs sont calculées suivant des coefficients  $s_i$  et  $p_i$  définis dans le TURPE.

L'ensemble de données  $D_1$  correspond à une grande usine dans l'industrie alimentaire avec une puissance souscrite typiquement dans la plage de 5000 à

	$D_1$	$D_2$
$(\underline{c}_1, \dots, \underline{c}_K)$	(2269, 2056, 72, 1769, 1152)	(1430, 1242, 1189, 910, 358)
$(\bar{c}_1, \dots, \bar{c}_K)$	(5536, 6732, 6667, 7539, 7189)	(2419, 2446, 2307, 1822, 1768)
$\mathbf{x}^0$	(5536, 6573, 6395, 6949, 6738)	(2419, 2425, 2254, 1730, 1691)
$\mathbf{x}^*$	(5536, 6479, 6479, 6914, 6914)	(2317, 2317, 2317, 2317, 2317)
	$D_3$	$D_4$
$(\underline{c}_1, \dots, \underline{c}_K)$	(533, 284, 208, 183, 2)	(480, 193, 321, 375, 315)
$(\bar{c}_1, \dots, \bar{c}_K)$	(1514, 1596, 1603, 1610, 1588)	(723, 737, 667, 722, 680)
$\mathbf{x}^0$	(1514, 1568, 1465, 1541, 1481)	(723, 736, 643, 700, 627)
$\mathbf{x}^*$	(1514, 1526, 1526, 1533, 1533)	(711, 711, 711, 711, 711)
	$D_5$	
$(\underline{c}_1, \dots, \underline{c}_K)$	(735, 545, 664, 572, 665)	
$(\bar{c}_1, \dots, \bar{c}_K)$	(1302, 1464, 1818, 2292, 1594)	
$\mathbf{x}^0$	(1302, 1449, 1238, 1716, 1449)	
$\mathbf{x}^*$	(1302, 1382, 1382, 1662, 1662)	

 TABLEAU 3.1 – Caractéristiques principales pour les jeux de données  $D_1$  à  $D_5$ 

7500kW ; l'ensemble de données  $D_2$  correspond à une entreprise de maintenance de matériel de transport avec une puissance souscrite typiquement dans la plage de 1500 à 2500kW ; l'ensemble de données  $D_3$  correspond à une boulangerie industrielle avec une puissance souscrite typiquement dans la plage de 1000 à 1500kW ; les jeux de données  $D_4$  et  $D_5$  correspondent à deux grands hôtels avec une puissance souscrite typiquement dans la plage de 500 à 1500kW.

Les principales caractéristiques de chaque ensemble de données sont présentées dans le [Tableau 3.1](#), à savoir pour chaque catégorie temporelle  $i$  : les valeurs minimales  $\underline{c}$  et les valeurs maximales  $\bar{c}$  des demandes  $c_t$  pour  $t \in T_i$  ; le minimum  $\mathbf{x}_i^0$  de  $F_i(x_i)$ , la composante de la fonction objectif correspondant à la catégorie de temps  $i$  ; les composantes  $\mathbf{x}_i^*$  de la solution optimale  $\mathbf{x}^*$  du PSCE.

Le [Tableau 3.1](#) suggère les premiers commentaires suivants :

- On observe naturellement que les 5 valeurs  $\mathbf{x}_i^0$  ne respectent pas forcément les contraintes d'ordre imposées par le PSCE, alors que les 5 valeurs  $\mathbf{x}_i^*$  satisfont toujours les contraintes d'ordre  $x_i \leq x_{i+1}$  ( $i = 1, \dots, 4$ ).
- Les valeurs minimales  $\underline{c}$  sont assez peu stables entre les  $K$  catégories temporelles. En effet, elles représentent des états où les machines sont *presque* toutes éteintes, mais parfois on retrouve des valeurs totalement aberrantes proche de 0kW (e.g. 2kW pour  $D_3$  et 72kW pour  $D_1$ ). Ces valeurs aberrantes peuvent être expliquées par des coupures de courant par exemple.

- Les valeurs maximales  $\bar{c}$  sont stables entre les  $K$  catégories temporelles, la différence entre les catégories temporelles est assez faible (de l'ordre de 10% et suivant la saison haute ou saison basse), mais parfois on retrouve des valeurs aberrantes (e.g. 1818kW et 2292kW pour  $D_5$ ) liés a un pics de consommation rare.
- Les composantes des solutions  $\mathbf{x}^0$  sont très proches des composantes  $\bar{c}_i$ , voir égale. Cela est surtout vrai pour  $i = 1$  (i.e. la pointe), car le très grand coefficient ne favorise pas du tout le dépassement de puissance souscrite.
- Les composantes des solutions  $\mathbf{x}$  sont assez proches des composantes  $\mathbf{x}^0$ . Quand plusieurs contraintes d'ordre consécutive  $I = \{\underline{i}, \underline{i} + 1, \dots, \bar{i}, \bar{i} + 1\}$  ne sont pas respectées par  $\mathbf{x}^0$ , alors les contraintes en question sont serrées (cf.  $D_2$  et  $D_4$ ). Ainsi les composantes en question sont égales à une valeur proche de  $\mathbf{x}_{\underline{i}}^0$ , car dans la formulation du TURPE on ne souhaite pas *trop* dépasser la puissance souscrite.
- On observe principalement deux types de solutions  $\mathbf{x}$ , celles où toutes les contraintes d'ordre sont actives et celles où seulement deux d'entre elles sont actives (correspondant à "heures pleines hiver est inférieure ou égale à heures creuses hiver" et "heures pleines été est inférieure ou égale à heures creuses été"). Les jeux de données ayant un profil hiver/été tel que la consommation d'hiver est supérieure à la consommation d'été (comme  $D_2$  et  $D_4$ ) vont plutôt favoriser une solution où toutes les contraintes d'ordre sont actives, tandis que les jeux de données ayant un profil hiver/été tel que la consommation d'hiver est inférieure à la consommation d'été et où la consommation durant la pointe (pour  $i = 1$ ) est contrôlée (comme  $D_1$ ,  $D_3$  et  $D_5$ ) vont plutôt favoriser une solution où deux des contraintes d'ordre sont actives.

Du fait de la présence des valeurs aberrantes interagissant sur  $\underline{c}_i$  et sur  $\bar{c}_i$ , donc sur  $\underline{c}$  et sur  $\bar{c}$ . Ainsi la valeur  $C = \bar{c} - \underline{c} + 1$ , qui intervient dans la complexité dans le pire des cas des algorithmes, n'est pas toujours très correct. Dans la [Sous-section 3.2.3](#) suivante, nous abordons le fait que l'algorithme `scaling_PAV` a une complexité en lien avec cette valeur  $C$ , alors qu'en réalité `OPTIM_SP` a une complexité en lien avec les différences  $c_i = \bar{c}_i - \underline{c}_i$  prisent séparément.

### 3.2.2 Calcul de complexité de OPTIM\_SP et scaling\_PAV

L'algorithme scaling\_PAV atteint une complexité dans le pire des cas de  $O(K \log(C))$  et l'algorithme OPTIM\_SP atteint la complexité dans le pire des cas de  $O(K^2 \log(C))$  (cf. Proposition 3.3). Cependant dans le cas de PSCE, le terme  $C$  des  $\log(C)$  intervient de manière assez différent.

Dans le cas de scaling\_PAV, il y a exactement  $K \log(C)$  itérations de recherche par dichotomie classique, où globalement durant les premières itérations il y a beaucoup plus d'opérations arithmétiques à faire que dans les dernières (cf. les courbes  $N_i^+(x_i)$  de la Figure 3.1). Durant les dernières itérations, si la solution est très proche de  $\bar{c}_i$ , alors il n'y a pratiquement aucune opération arithmétique à faire, sinon il y a un certain nombre d'opérations arithmétiques à faire et ce nombre est assez stable (car la fonction  $F$  est évaluée sur un nombre de valeurs similaires).

Dans le cas de OPTIM\_SP, pour la résolution de  $\mathcal{P}^0$ , il y a aussi  $O(K \log(C))$  itérations de recherche par dichotomie, mais de A\_SEARCH entre  $\underline{c}_i$  et  $\bar{c}_i$  avec un certain paramètre  $a$ . En principe, durant la première itération, il y a beaucoup d'opérations arithmétiques à faire, mais moins que ceux d'une recherche par dichotomie classique, car le point intermédiaire choisit est plus grand. Dès la deuxième itération, la valeur  $ymid$  évaluée sur  $F$  est plutôt 'proche' de l'optimale, nécessitant donc pas ou peu d'opérations arithmétiques. Pour la résolution des itérations pour  $\mathcal{P}^k$  pour  $1 \leq k \leq K - 1$ , qui nécessitent un total de  $O(K^2 \log(C))$  appels à A\_SEARCH dans le pire cas, sont en réalité beaucoup plus rapide. Dans la pratique, l'espace de recherche est en moyenne plus petit car la solution  $yopt$  de A\_SEARCH est toujours entre les deux valeurs  $\mathbf{x}_i$  et  $\mathbf{x}_{i+1}$  violant la contrainte d'ordre  $x_i \leq x_{i+1}$ . De plus,  $\mathbf{x}_i$  et  $\mathbf{x}_{i+1}$  sont souvent de grandes valeurs, il faut donc très peu d'opérations arithmétiques pour trouver le minimum  $yopt$  des A\_SEARCH résolvant les problèmes  $\mathcal{P}^k$ . C'est pourquoi la phase des itérations  $1 \leq k \leq K - 1$  est très rapide.

### 3.2.3 Évaluation comparative de OPTIM\_SP et scaling\_PAV

Tous les tests rapportés dans le Tableau 3.2 ont été réalisés en utilisant Python 3.6.8 sur un processeur quadricœur de 8-threads avec 16 Go de RAM de 2,8 GHz fonctionnant sous Windows 10 (64 bits).

	$D_1$			$D_2$		
$a$	#evalF	Total_N <sup>+</sup>	cpu	#evalF	Total_N <sup>+</sup>	cpu
sPAV	130	113117	44	110	62766	28
1	192	103498	50	253	72739	34
2	174	39913	25	256	29724	15
3	172	18923	10	234	14463	8
4	167	11286	6	246	12082	8
5	166	7754	5	252	15993	9
6	183	5929	4	248	13183	8
7	168	4801	5	257	11984	7
8	169	4037	5	255	10989	7
	$D_3$			$D_4$		
$a$	#evalF	Total_N <sup>+</sup>	cpu	#evalF	Total_N <sup>+</sup>	cpu
sPAV	110	101162	38	90	93442	41
1	162	90373	44	199	77807	37
2	149	49375	24	171	34229	17
3	149	24729	11	169	18330	9
4	144	11882	6	164	9760	5
5	134	6426	4	162	6597	4
6	139	4248	3	159	22971	11
7	146	3434	3	160	20586	11
8	147	2977	2	174	19567	11
	$D_5$					
$a$	#evalF	Total_N <sup>+</sup>	cpu			
sPAV	110	15022	7			
1	167	23838	12			
2	162	24938	12			
3	155	14982	8			
4	161	11746	6			
5	157	9481	5			
6	160	24505	13			
7	167	22706	12			
8	167	21914	11			

TABLEAU 3.2 – Résultats d'exécution pour les jeux de données  $D_1, \dots, D_5$

Ce tableau présente les résultats de calcul obtenus avec `scaling_PAV` (première ligne du tableau intitulée 'sPAV') et `OPTIM_SP` (dans les lignes suivantes pour des valeurs de  $a$  allant de 1 à 8) pour les différentes instances correspondant aux jeux de données  $D_1$  à  $D_5$ . Pour chaque instance, la colonne intitulée '#evalF' indique le nombre d'évaluation des terme de l'objectif; la colonne intitulée 'Total\_N+' indique l'évaluation du nombre total d'opérations arithmétiques résultant, mesuré à l'aide des valeurs  $N_i^+(x_i)$ ; la colonne intitulée 'cpu' indique le temps cpu total résultant (en millisecondes).

Rappelons que plus  $x_i$  est grand, alors plus  $N_i^+(x_i)$  sera petit (cf. [Figure 3.1](#)). Ainsi lorsque peu d'évaluation  $F_i(x)$  sont faites sur des petits valeurs, alors le Total\_N+ se voit faible. Au contraire, plus il y a d'évaluation avec de petites valeurs, alors plus le Total\_N+ se voit grand.

Les résultats présentés dans le [Tableau 3.2](#) suggèrent les commentaires suivants :

- Pour  $D_1$  à  $D_4$ , l'algorithme `OPTIM_SP` surpasse significativement `scaling_PAV` en termes de temps CPU pour  $a$  choisi entre 4 et 8, et la comparaison reste favorable dans l'intervalle plus large  $2 \leq a \leq 8$ . Clairement,  $\bar{c}$  et  $\mathbf{x}^*$  sont assez proches (cf. [Tableau 3.1](#)), ce qui favorise donc une recherche avec  $a > 1$ , afin de ne pas du tout explorer les petites valeurs.
- Pour  $D_5$ , on peut observer qu'il existe seulement deux valeurs de  $a$  pour lesquelles `OPTIM_SP` surpasse `scaling_PAV`. En effet, `scaling_PAV` est beaucoup plus sensible aux valeurs aberrantes, qui peuvent énormément modifier  $C$ , ainsi il peut arriver exceptionnellement qu'il soit extrêmement rapide, e.g. la valeur 2292 de  $D_5$  (cf. [Tableau 3.1](#)) permet de choisir des points intermédiaires de recherche par dichotomie très proche du minimum dès la première itération. D'autre part, cette valeur aberrante force les résolutions de `A_SEARCH` pour `OPTIM_SP` à converger vers la solution par les faibles valeurs, e.g. ainsi pour chaque point intermédiaire l'évaluation des terme de l'objectif va nécessiter un grand nombre d'opérations arithmétiques.
- Dans la plupart des cas, on peut observer qu'il existe une plage assez large de valeurs de  $a$  pour lesquelles `OPTIM_SP` atteint une efficacité améliorée par rapport à l'un des meilleurs algorithmes connus précédemment pour résoudre les problèmes de minimisation d'une fonction convexe à objectif séparable sous contraintes d'ordre. Cela montre que l'algorithme `OPTIM_SP` présente une bonne robustesse par rapport au paramètre  $a$ , et que le choix

d'une valeur particulière pour le paramètre  $a$  n'est pas un problème critique : une conséquence pratique de ceci est qu'une bonne valeur pour  $a$  sera facilement obtenue sur une base expérimentale, en observant le comportement de l'algorithme `OPTIM_SP` sur un échantillon de quelques instances typiques du problème pour quelques valeurs de  $a$  dans l'intervalle  $[2, 6]$ .

- Pour chaque résultat présenté dans le [Tableau 3.2](#), si on calcule le rapport  $\text{cpu}/\text{TotalN}^+$ , alors on observerait que toutes les valeurs obtenues sont typiquement comprises entre  $5e-4$  et  $7e-4$ , ce qui montre que le `cpu` est, à de petite fluctuation près, proportionnel à  $\text{TotalN}^+$ . Ceci confirme la pertinence de notre analyse basée sur l'utilisation de  $\text{TotalN}^+$  comme mesure de l'effort de calcul.

### 3.3 Résumé du chapitre

Nous proposons un algorithme efficace en temps polynomial basé sur une approche d'ensemble de contraintes actives, qu'on appelle `OPTIM_SP`, pour la minimisation d'un problème convexe à objectif séparable. Les résultats de calcul obtenus montrent que l'algorithme présente un temps de calcul amélioré par rapport à `scaling_PAV` [3], l'une des meilleures méthodes existantes pour ce problème. En effet, dans le cadre du PSCE, l'évaluation de la fonction objectif ne se fait pas en  $O(1)$ , ainsi `OPTIM_SP` est spécialement conçu pour réduire l'effort de calcul sur des données historiques à grande échelle. Pour minimiser une fonction convexe, on utilise une recherche par dichotomie ajustable qui prend en argument un certain paramètre  $a$  afin de diminuer l'effort de calcul pour chacune de ces évaluations.



# Chapitre 4

## Problème de souscription de contrat d'électricité avec des incertitudes sur les consommations

### Sommaire

---

4.1	Bref rappel du problème . . . . .	52
4.2	Modèle robuste du PSCE . . . . .	53
4.2.1	Ensemble d'incertitude . . . . .	53
4.2.2	Formulation . . . . .	54
4.3	Résolution de la version robuste du PSCE . . . . .	56
4.4	Résultats expérimentaux . . . . .	58
4.4.1	Analyse de l'ensemble d'incertitude . . . . .	58
4.4.2	Résultats numériques concernant la version robuste . . . . .	59
4.4.3	Évaluation comparative de OPTIM_SP et scaling_PAV . . . . .	61
4.5	Résumé du chapitre . . . . .	61

---

Nous abordons dans ce chapitre une version robuste du problème de souscription des contrats d'électricité (PSCE) du [Chapitre 4](#). On applique une combinaison de la décomposition *Seasonal and Trend decomposition using Loess* (STL) de Cleveland, R. B., Cleveland, W. S., McRae, J. E., et Terpenning, I. (1990) [31] et un concept d'ensemble d'incertitude similaire à celui de Bertsimas, D., et Sim, M. (2004) [15].

Par chance, la version robuste du PSCE peut être résolue efficacement en utilisant les mêmes algorithmes que pour la version déterministe e.g. OPTIM\_SP. Cependant, la prise en compte de l'incertitude nécessite le calcul des valeurs les plus défavorables de la fonction objectif, ce dernier problème est réduit à la résolution d'un problème de sac à dos à choix multiples, qui est résoluble en temps polynomiale avec Zemel, E. (1984) [108]. Finalement, la résolution de la version robuste du PSCE se résout donc en temps polynomial.

## 4.1 Bref rappel du problème

Le problème consiste à trouver les puissances souscrites d'un contrat d'électricité pour un client en fonction de son historique de consommation d'électricité sur une période de temps fixe. L'objectif est d'optimiser la facture d'électricité composée du coût de la fourniture, du coût de la souscription du contrat d'électricité et des pénalités dues aux dépassements de puissance souscrite et de certain coût fixe. Le PSCE, dans sa version déterministe, peut alors être formulé comme le programme mathématique suivant  $\mathcal{P}$  :

$$\mathcal{P} : \min F(x) = \sum_{i=1}^K F_i(x_i) \quad (2.2)$$

$$s.t. \quad x_i \leq x_{i+1}, \quad 1 \leq i \leq K-1, \quad (2.3)$$

$$\underline{c} \leq x_i \leq \bar{c}, \quad 1 \leq i \leq K, \quad (2.4)$$

La fonction objectif Équation (2.2) est convexe et séparable comme l'indique la Proposition 2.1. Les contraintes Équation (2.3) sont appelées *les contraintes d'ordre* et les contraintes Équation (2.4) sont appelées *les contraintes de borne*. Une solution  $\mathbf{x} = (\mathbf{x}_1, \dots, \mathbf{x}_K)^T$  pour  $\mathcal{P}$  est un vecteur de dimensions  $K$  satisfaisant les contraintes Équation (2.3) et Équation (2.4).

Le PSCE peut être formulé comme un problème d'optimisation convexe à objectif séparable soumis à des contraintes d'ordre total, et se résout efficacement avec l'algorithme OPTIM\_SP basé sur une approche d'ensemble de contraintes actives. L'algorithme résout une série de problèmes intermédiaires du PSCE en jouant sur le nombre de contraintes. Pour chacun des problèmes intermédiaires, on utilise la solution du problème précédent dans une série de recherche par dichotomie ajustable, qui prend en argument un certain paramètre  $a$ . Ce paramètre

$a$  permet de diminuer le nombre d'évaluation de la fonction objectif (partielle ou totale) mais aussi diminuer l'effort de calcul durant ces évaluations.

## 4.2 Modèle robuste du PSCE

### 4.2.1 Ensemble d'incertitude

La consommation est considérée comme une série temporelle dans laquelle chaque échantillon est collecté toutes les 10 minutes (52560 mesures par an). Nous appliquons la méthode STL [31] pour décomposer la série temporelle afin de récupérer la donnée de consommation traitée et les informations du résiduel (cf. [Algorithme 4.1](#)). Dans notre application, nous découpons l'ensemble de donnée brute à disposition en 5 séries temporelles (une par catégorie temporelle), puis nous appliquons une décomposition STL sur chacune des séries temporelles.

Pour chaque catégorie, la donnée traitée  $c_t$  est obtenue en sommant la composante de la tendance et la composante de la saisonnalité, e.g.  $c_t = trend(t) + seasonal(t)$ , réduit à l'horizon voulu ; puis on regroupe les données traitées des 5 catégories temporelles pour construire la donnée traitée finale. D'une certaine manière, on peut dire que la donnée traitée est "nettoyée" du bruit. On peut aussi travailler avec des prédictions de  $trend(t)$  et/ou de  $seasonal(t)$ , afin de récupérer des consommations prédites.

La composante résiduelle *remainder* peut être considérée comme les réalisations d'une variable aléatoire. En général, le résiduel correspond à des réalisations indépendantes d'une certaine distribution de probabilité connue (typiquement une distribution normale ou une distribution normale tronquée) avec une moyenne nulle. Ainsi, on peut considérer que 99.7% des données brutes tombent entre  $c_t - 3\sigma$  et  $c_t + 3\sigma$ , tel que  $\sigma$  soit l'écart type de *remainder*. Si le résiduel ne correspond pas à des réalisations de ce type, alors on peut découper l'horizon en morceaux plus petit permettant de mieux respecter ces propriétés, e.g. au lieu de découper suivant les 5 catégories, on peut ajouter un sous-découpage supplémentaire par mois. Finalement, on peut s'imaginer travailler sur des données traitées  $c_t$  autorisant une petite variation qui est comprise entre  $-3\sigma$  et  $3\sigma$ .

Dans le problème du PSCE robuste, la décomposition STL sur chacune des catégories temporelles prises séparément permet d'une part de pouvoir définir une saisonnalité de longueur différente sur l'horizon, entre les heures pleines hiver

( $i = 2$ ) et les heures pleines été ( $i = 4$ ) i.e. 12 heures contre 16 heures par jour ; cela permet d'autre part d'obtenir des écarts-types plus précis (et plus faibles) qu'on note  $\sigma_i$ .

Pour rappel, on souhaite définir un ensemble d'incertitude similaire à l'ensemble d'incertitude de type budget d'incertitude, où les paramètres incertains prennent une valeur dans un certain intervalle et où un budget d'incertitude contrôle le nombre de paramètres autorisés à dévier de leurs valeurs nominales.

On note  $v^i = (v_1^i, \dots, v_{|T_i|}^i)$  le vecteur des variables de déviations. Soient  $b_i$  la variation maximale autorisée. Lorsque le *residual* est de bonne qualité, on peut ainsi borner les déviations par  $-b_i \leq v_t^i \leq b_i$  tel que  $b_i = r_i * \sigma_i$  et  $r_i = 3$ . Ainsi, on reproduit jusqu'à 99.7% des réalisations possibles de *residual*. Soient  $B_i$  le budget d'incertitude. Les variations totales doivent satisfaire la contrainte  $\sum_{t=1}^{T_i} |v_t^i| \leq B_i$ . À titre d'indication sur la manière de choisir  $B_i$ , considérons  $n$  réalisations d'une variable aléatoire normalement distribuée avec une moyenne nulle et un écart-type  $\sigma_i$ . La somme de ces  $n$  réalisations aléatoires a une variance  $\sigma^2 = n\sigma_i^2$ , c'est-à-dire que son écart-type est  $\sigma = \sqrt{n}\sigma_i$ . Ainsi  $B_i = R_i * b_i$  tel que  $R_i$  est le nombre entier de paramètres incertains qu'on autorise à varier et est du même ordre de grandeur que  $\lfloor \sqrt{T_i} \rfloor$  (avec  $T_i$  le nombre de pas de temps dans la série temporelle).

L'ensemble d'incertitude  $U_i$  est alors défini comme :

$$U_i = \{v^i \mid -b_i \leq v_t^i \leq b_i, \forall t, \sum_{t=1}^{T_i} |v_t^i| \leq B_i\}. \quad (4.1)$$

**Code python pour STL** Un exemple de code python, pour appliquer la décomposition STL et récupérer les résultats, est montré dans [Algorithme 4.1](#). La fonction prend en paramètre le jeu de donnée sous forme de DataFrame et la période de la saisonnalité (suivant le problème : quotidien ou hebdomadaire ou mensuelle).

## 4.2.2 Formulation

Pour chaque catégorie  $i$ , la quantité dépassée pour la période  $t$  est maintenant  $\delta_t^r(x_i, v_t^i) = \max(0, (c_t + v_t^i) - x_i)$ , notons que  $\delta_t^r(x_i, v_t^i) = \delta_t(x_i - v_t^i)$ . Pour la catégorie  $i$  et pour un  $x_i$  donnés, le coût de dépassement le plus défavorable dans la version robuste du problème est obtenu comme la valeur maximale sur l'ensemble

**Algorithme 4.1** Code python pour l'utilisation de STL

```

1  import numpy as np
2  from statsmodels.tsa.seasonal import STL
3
4  def applicable_robustness(df, period):
5      stl = STL(df.conso, period = period)
6      res = stl.fit()
7
8      data = (res.trend + res.seasonal).to_dict().values()
9      resid = (np.mean(res.resid), np.std(res.resid))
10
11     return data, resid

```

d'incertitude  $U_i$  de la fonction de coût de pénalité, qui est définie comme :

$$\Pi_i^r(x_i) = \max_{v^i \in U_i} \sum_m \sqrt{\sum_{t \in T_{i,m}} (\delta_t^r(x_i, v_t^i))^2}. \quad (4.2)$$

La fonction objectif pour la formulation robuste est  $F^r(x) = \sum_i F_i^r(x_i) = \sum_i s_i x_i + p_i \Pi_i^r(x_i)$  et la version robuste du PSCE peut être énoncée comme :

$$\begin{aligned} \mathcal{P}^r : \quad & \min F^r(x) \\ \text{s.t.} \quad & x_i \leq x_{i+1}, \quad 1 \leq i \leq K-1, \\ & \underline{c} \leq x_i \leq \max_{1 \leq i' \leq K} \{\bar{c}_{i'} + b_{i'}\}, \quad 1 \leq i \leq K. \end{aligned}$$

Notons que dans la formulation ci-dessus, les valeurs des bornes supérieures des variables  $x_i$  ont été modifiées pour tenir compte de l'augmentation possible de la valeur maximale de la consommation de  $b_i$  pour la catégorie  $i$ .

**Proposition 4.1.** *Chaque  $\Pi_i^r(x_i)$  est convexe en  $x_i$ .*

*Démonstration.* Pour tout  $v^i \in U_i$  donné, nous savons de la [Proposition 2.1](#) que la fonction suivante est convexe en  $x_i$  :

$$\sum_m \sqrt{\sum_{t \in T_{i,m}} (\delta_t^r(x_i, v_t^i))^2}.$$

Maintenant, puisque la valeur maximale dans [Équation \(4.2\)](#) doit être déterminée par rapport à l'ensemble (fini) des points extrêmes de  $U_i$ ,  $\Pi_i^r$  est le maximum

ponctuel d'une collection finie de fonctions convexe en  $x_i$ . Ceci prouve le résultat affirmé.  $\square$

**Corollaire 4.2.** *Chaque  $F_i^r$  est convexe en  $x_i$ , et donc  $\mathcal{P}^r$  est un programme convexe.*

### 4.3 Résolution de la version robuste du PSCE

Puisqu'il apparaît que la version robuste du PSCE a la même structure que la version déterministe, l'algorithme OPTIM\_SP s'avère être facilement applicable pour la version robuste du PSCE. Cependant pour un ensemble de catégorie  $I$  donné, chaque itération de la fonction A\_SEARCH est associée à l'évaluation de  $F_I^r(x_I) = \sum_{i \in I} F_i^r(x_I)$  pour un  $x_I$  donné, ce qui est équivalent à la résolution de  $|I|$  sous-problèmes [Équation \(4.2\)](#).

**Proposition 4.3.** *Pour un  $x_i$  donné, toute solution optimale du sous-problème [Équation \(4.2\)](#) présente  $R_i$  composantes égales à  $b_i$ , et toutes les autres composantes étant égales à 0.*

*Démonstration.* Pour une catégorie  $i$  et un  $x_i$  donnés, l'objectif du sous-problème [Équation \(4.2\)](#) est de maximiser une fonction convexe en fonction de  $v^i$ , et il est bien connu que le maximum est atteint en un point extrême du polyèdre représentant l'ensemble d'incertitude  $U_i$ . Puisque  $R_i$  est supposé être une valeur entière, chaque point extrême du polyèdre  $U_i$  comporte exactement  $R_i$  composantes non nulles égales à leurs valeurs de borne supérieure  $b_i$ .  $\square$

Pour une catégorie  $i$  et pour un  $x_i$  donnés, selon la [Proposition 4.3](#), le sous-problème de maximisation [Équation \(4.2\)](#) est résolu lorsque les  $R_i$  paramètres incertains qui peuvent augmenter le plus la fonction objectif sont trouvés. C'est-à-dire que nous devons trouver comment choisir ces  $R_i$  pas de temps sur les  $M_i$  mois de la catégorie  $i$ . Ce problème de choix optimale, [Équation \(4.2\)](#) peut être reformulé en un *problème de sac à dos à choix multiples* (PSDCM) avec des variables entières. Ce problème peut être résolu avec une complexité dans le pire des cas de  $O(N)$ , où  $N = R_i * |M_i|$  est le nombre d'articles, avec un algorithme se basant sur la convexité du problème dual [108]. Ce PSDCM peut être formulé en utilisant des variables binaires  $z_{m,n}$  pour  $m \in M_i$  et  $n \in \{0, \dots, R_i\}$ , où si  $z_{m,n} = 1$ , alors l'algorithme attribue  $n$  écarts au mois  $m$ . Soit  $T_{i,m}^n \subseteq T_{i,m}$  le sous-ensemble correspondant aux  $n$  plus grandes valeurs de  $c_t$  de  $T_{i,m}$ .

**Proposition 4.4.** *Pour une catégorie  $i$  et un mois  $m$  donnés, si exactement  $n$  paramètres incertains  $v_t^i$ , pour  $t \in T_{i,m}$ , peuvent prendre leur borne supérieure  $b_i$ , alors le pire cas pour ce mois  $m$  est obtenue lorsque  $v_t^i = b_i$  pour les  $t$  appartenant à  $T_{i,m}^n$ .*

*Démonstration.* Pour une catégorie  $i$  et un mois  $m$  donnés, la pire augmentation possible de la fonction de coût de pénalité dépend uniquement des termes  $\delta_t^r(x_i)^2$ . Ainsi on obtient le pire en augmentant la valeur sous le carré des plus grandes valeurs. Finalement, on cherche à souhaiter que les déviations des  $n$  périodes  $t$  ayant les  $\delta_t^r$  les plus élevés soient fixées à la borne  $b_i$ .  $\square$

Selon la [Proposition 4.4](#), lorsque  $n$  paramètres sont fixés égaux à leur borne supérieure  $b_i$ , le coût de pénalité pour le mois  $m$  est :

$$\gamma_{m,n} = \sqrt{\sum_{t \in T_{i,m}^n} (\delta_t^r(x_i, b_i))^2 + \sum_{t \in T_{i,m} \setminus T_{i,m}^n} (\delta_t^r(x_i))^2}.$$

Le PSDCM à résoudre pour déterminer le coût du pire cas est alors le suivant :

$$\text{PSDCM : } \max \sum_{m=1}^{M_i} \sum_{n=0}^{R_i} z_{m,n} \gamma_{m,n} \quad (4.3)$$

$$\text{s.t. } \sum_{m=1}^{M_i} \sum_{n=0}^{R_i} n z_{m,n} \leq R_i, \quad (4.4)$$

$$\sum_{n=0}^{R_i} z_{m,n} \leq 1, \quad \forall m, \quad (4.5)$$

$$z_{m,n} \in \{0, 1\}, \forall m \forall n. \quad (4.6)$$

La fonction objectif [Équation \(4.3\)](#) est le coût de la dépassement du pire cas de la catégorie  $i$ , soit l'équivalent de [Équation \(4.2\)](#). La contrainte [Équation \(4.4\)](#) assure que le budget d'incertitude est vérifié. La contrainte [Équation \(4.5\)](#) assure qu'il n'y a bien qu'un seul booléen  $z_{m,n}$  qui peut prendre la valeur 1 pour chaque mois.

**Complexité** Rappelons que  $M_i \subseteq M$  et  $R_i$  est une valeur entière du même ordre de grandeur que  $\sqrt{T_i}$  (où  $T_i \leq T$ ), ainsi pour un  $x_i$  donné, déterminer la valeur optimale de [Équation \(4.2\)](#) avec une complexité dans le pire des cas de  $O(|M| * \sqrt{T})$ . Par conséquent, nous pouvons énoncer la [Proposition 4.5](#) suivante :

Catégorie	Mois	$\underline{c}_i$	$\bar{c}_i$	$T_i$	$\lfloor \sqrt{T_i} \rfloor$	$\sigma_i$
Pointe	Dec-Fev	2269	5536	1488	38	115
Heures Pleines Hiver	Nov-Mar	2056	6732	8688	93	124
Heures Creuses Hiver	Nov-Mar	72	6667	11562	107	104
Heures Pleines Été	Avr-Oct	1769	7539	13920	117	144
Heures Creuses Été	Avr-Oct	1152	7189	16902	130	134

TABLEAU 4.1 – Caractéristiques principales de  $D_1$  pour le problème robuste

**Proposition 4.5.** *La complexité dans le pire des cas de OPTIM\_SP appliquée à la version robuste du PSCE est de  $O(K^2|M|\sqrt{T}\log C)$ .*

## 4.4 Résultats expérimentaux

Cette section présente diverses expériences réalisées sur la version robuste du PSCE pour l'instance  $D_1$  du Chapitre 4. Notons que des résultats similaires sont observés pour les autres ensembles de données.

### 4.4.1 Analyse de l'ensemble d'incertitude

Les principales caractéristiques du jeu de données pour le problème sont présentées dans le tableau 4.1, à savoir pour chaque catégorie temporelle : les mois en question ; les valeurs minimales  $\underline{c}_i$  et les valeurs maximales  $\bar{c}_i$  des consommations  $c_t$  ;  $T_i$ , le nombre de données et  $\lfloor \sqrt{T_i} \rfloor$ , la racine carrée de ce nombre ; l'écart type du résiduel de la décomposition STL. Par exemple, les Heures Creuses Été couvrent les mois de Avril à Octobre, a  $T_i = 16902$  pas de temps de 10 minutes, et l'écart type du reste est  $\sigma_i = 134$ .

**Choix des  $b_i$**  Les tests statistiques sur les résiduels des  $K$  décompositions STL révèlent qu'ils sont non corrélés, ont des moyennes nulles et sont normalement distribués. C'est pourquoi, en considérant ces résiduels comme des distributions normales tronquées entre  $-r_i\sigma_i$  et  $r_i\sigma_i$  avec  $r_i = 3$ , on obtient une représentation correcte de l'écart de la consommation. Le paramètre robuste  $b_i$  est donc choisi pour être  $3 * \sigma_i$ . Pour les heures creuses d'été, cela signifie qu'on autorisera certaines consommations  $c_t$  d'augmenter de la valeur 402. À titre d'indication, la consommation valant 7189kW devient 7591kW soit 5.5% de plus.

q	Subscribed Power (SP)	Cost	cpu
0	6200, 7000, 7000, 7300, 7300	112275	
0.2	6200, 7000, 7000, 7300, 7300	114274	
0.5	6200, 7000, 7000, 7300, 7300	114315	
1	6200, 7000, 7000, 7300, 7300	114315	
0	5536, 6479, 6479, 7316, 7316	107576	25
0.2	5881, 6837, 6837, 7287, 7287	113750	2235
0.5	5881, 6837, 6837, 7341, 7341	113771	2404
1	5881, 6837, 6837, 7341, 7341	113771	2755
0	5881, 6837, 6837, 7341, 7341	111296	
0.2	5536, 6479, 6479, 6914, 6914	118212	
0.5	5536, 6479, 6479, 6914, 6914	121448	
1	5536, 6479, 6479, 6914, 6914	123161	

TABLEAU 4.2 – Comparaison des coûts de diverses solutions pour l’ensemble de données  $D_1$

**Choix des  $B_i$**  La définition du PSCE par le TURPE ne favorise pas un très grand nombre de dépassements, c’est pourquoi on fixe  $R_i = \lfloor q * \sqrt{T_i} \rfloor$  avec  $q \in \{0, 0.2, 0.5, 1\}$ . On remarque que pour  $q = 0$  on obtient la version déterministe du problème. Pour les heures creuses d’été, les différentes valeurs de  $q$  considérées correspondent à des valeurs de  $R_i$  valant 0, 26, 65 et 130, et cela signifie qu’on autorisera par exemple  $\frac{130}{16902} = 0.77\%$  des pas de temps à augmenter leurs consommations  $c_t$  de la valeur  $b_i$ . Suivant [Équation \(4.1\)](#), on prend  $B_i = R_i b_i$ .

#### 4.4.2 Résultats numériques concernant la version robuste

Le [Tableau 4.2](#) fournit des informations comparatives sur les coûts de diverses solutions optimales ou sous-optimales pour la version déterministe ( $q = 0$ ) et la version robuste ( $q > 0$ ) du PSCE pour l’ensemble de données  $D_1$ . La colonne intitulée ‘q’ indique niveau de robustesse ; la colonne intitulée ‘Subscribed Power’ indique le vecteur solution ; la colonne intitulée ‘Cost’ indique le coût de la solution SP pour niveau de robustesse  $q$  ; la colonne intitulée ‘cpu’ indique le temps cpu total résultant (en millisecondes).

Le [Tableau 4.2](#) se décompose en trois parties. La première partie du tableau concerne le contrat actuellement utilisé par le client (une grande usine de l’industrie alimentaire), la solution correspondante (6200, 7000, 7000, 7300, 7300), est appelée solution de référence ; le coût en termes de fonction objectif robuste pour

$q = 0, 2, 0, 5, 1$  sont fournis. La deuxième partie concerne les solutions optimales de la version déterministe (pour  $q = 0$ ) et des versions robustes du PSCE (pour  $q = 0.2, 0.5, 1$ ). La troisième partie présente le coût déterministe 111296 de la solution optimale robuste obtenue pour  $q = 0.5$  et  $q = 1$ , à savoir : (5881, 6837, 6837, 7341, 7341) ; les valeurs de la fonction objectif robuste pour  $q = 0.2, 0.5, 1$  de la solution déterministe optimale (5536, 6479, 6479, 6914, 6914).

Les résultats du [Tableau 4.2](#) suggèrent les commentaires suivants :

- La comparaison des coûts des solutions optimales (deuxième partie du tableau) avec les coûts de la solution de référence montre que cette dernière est sous-optimale dans tous les cas : la différence est de 4,2% pour  $q = 0$ , 0,4% pour  $q = 0.2$ , et 0,5% pour  $q = 0.5$  et  $q = 1$ . On constate donc que la solution de référence s'avère être une assez bonne approximation (à 0,5% près) des solutions robustes optimales pour toutes les valeurs (strictement positives) de  $q$  considérées.
- La comparaison entre les trois dernières lignes de la deuxième partie et les trois dernières lignes de la troisième partie montre le gain apporté par la solution optimale robuste par rapport à la solution optimale déterministe en présence d'incertitude : lorsque  $q = 0.2$ , la valeur de la fonction robuste optimale est de 113750, alors que la valeur de la fonction objectif robuste pour la solution optimale déterministe est de 118212, montrant que la première conduit à une amélioration du coût de 3.8% par rapport à la seconde. La même comparaison pour  $q = 0.5$  et  $q = 1$  conduirait à une amélioration respective de 6.3% et 7.6%.
- La comparaison entre la première ligne de la deuxième partie du tableau et la première ligne de la troisième partie montre que la différence entre la valeur de la fonction objectif déterministe de la solution robuste optimale pour  $q = 0.5$  et  $q = 1$ , et la valeur de la solution déterministe optimale (111296-107576= 3720) représente une augmentation de 3.3%. Cette valeur peut être interprétée comme le prix de la robustesse.
- Les temps de calcul nécessaires pour résoudre les versions robustes du PSCE sont environ 100 fois plus élevés que pour résoudre la version déterministe, mais ils ne dépassent généralement pas deux ou trois secondes, ce qui est tout à fait acceptable du point de vue de l'applicabilité pratique.

### 4.4.3 Évaluation comparative de OPTIM\_SP et scaling\_PAV

En plus des résultats discutés dans la [Sous-section 4.4.2](#), nous fournissons dans le [Tableau 4.3](#) des résultats montrant l'efficacité de calcul de OPTIM\_SP appliqué à la version robuste du PSCE, sous une forme similaire au [Tableau 3.2](#). Dans ces expériences, la valeur  $q = 0.5$  a été choisie. Les principaux commentaires suggérés par les résultats présentés dans le [Tableau 4.3](#) sont les suivants :

- Étant donné que l'effort de calcul nécessaire pour calculer la valeur de chaque composante de la fonction objectif du problème robuste est nettement plus important que pour le cas déterministe, il n'est pas surprenant d'observer des valeurs de cpu nettement plus élevées. En moyenne, on peut voir qu'elles sont multipliées par un facteur de l'ordre de 50 à 100 (cf. [Tableau 3.2](#)).
- Pour ce qui concerne l'influence de la variation de la valeur du paramètre  $a$  dans l'intervalle  $a = 1$  à  $a = 8$ , on remarque un comportement similaire à celui observé pour le cas déterministe : pour toutes les instances (sauf  $D_5$ ), OPTIM\_SP surpasse Scaling\_PAV pour une large gamme de valeurs pour  $a$ .
- La relation entre le cpu et la mesure Total\_N<sup>+</sup> de l'effort de calcul, qui était presque linéaire dans le cas déterministe, est plus complexe dans le cas du PSCE robuste ; ceci est principalement dû à la surcharge significative induite par la nécessité de résoudre de manière répétée le PSDCM discuté dans la [Section 4.3](#).

## 4.5 Résumé du chapitre

Nous proposons une version robuste du problème de souscription des contrats d'électricité (PSCE) en utilisant une combinaison de la décomposition *Seasonal and Trend decomposition using Loess* [31] et un concept d'ensemble d'incertitude similaire à celui de [15]. Le problème consiste donc à trouver les meilleurs paramètres d'un contrat d'électricité pour un client en fonction de son historique de consommation d'électricité pouvant être soumise à différentes sources d'incertitude.

Il a été démontré que ce problème appartient à la même classe de problème, convexe à objectif séparable sous des contraintes d'ordre total, que sa version

	D1			D2		
$a$	#evalF	Total_N <sup>+</sup>	cpu	#evalF	Total_N <sup>+</sup>	cpu
sPAV	130	5733875	3815	110	2781354	2363
1	198	4786268	4058	213	3537792	3271
2	173	1398527	2046	183	1067315	1803
3	181	490332	1534	180	461635	1578
4	173	223949	1401	176	302938	1406
5	170	130601	1254	176	178908	1362
6	182	82507	1327	196	324828	1559
7	175	55507	1267	185	274221	1457
8	172	50018	1255	183	243549	1472

	D3			D4		
$a$	#evalF	Total_N <sup>+</sup>	cpu	#evalF	Total_N <sup>+</sup>	cpu
sPAV	110	4399366	3024	100	3949268	2834
1	163	4121953	3512	201	3074664	2971
2	148	1900657	2237	184	949165	1832
3	152	570448	1413	174	295506	1387
4	151	202383	1282	174	129488	1325
5	143	106005	1099	172	76850	1279
6	144	64673	1076	175	56555	1290
7	139	45094	1036	177	916144	1747
8	145	39596	1065	185	864616	1757

	D5		
$a$	#evalF	Total_N <sup>+</sup>	cpu
sPAV	110	539235	1115
1	171	825202	1695
2	160	171727	1273
3	156	643125	1472
4	156	432818	1388
5	161	355929	1398
6	166	1192940	1930
7	163	1078631	1810
8	167	997721	1773

TABLEAU 4.3 – Résultats d'exécution pour la version robuste du PSCE pour les jeux de données  $D_1, \dots, D_5$

déterministe, et peut être résolu efficacement en utilisant le même algorithme polynomial, e.g. OPTIM\_SP. Cependant, la prise en compte de l'ensemble d'incertitude nécessite le calcul du pire cas à chaque évaluation de la fonction objectif. Par chance, ces évaluations se traduisent par un problème de maximisation, où la solution est un point extrême du polyèdre représenté par l'ensemble d'incertitude. Le pire cas peut être ainsi déterminé par la résolution d'un problème de sac à dos à choix multiples, permettant de répartir le budget d'incertitude sur parmi les paramètres incertains. Ce sous-problème est résoluble en temps polynomial [108], ainsi le problème robuste du PSCE peut aussi être résolu en temps polynomial. Le choix des niveaux de robustesse est discuté et des résultats numériques y sont montrés.

CHAPITRE 4. PSCE AVEC INCERTITUDES SUR LES CONSOMMATIONS

## Deuxième partie

# Problème de gestion optimale des réseaux d'eau



# Chapitre 5

## Modèle pour la minimisation du coût de l'énergie avec RFLATS dans les réseaux d'eau ruraux

### Sommaire

---

5.1	Introduction . . . . .	68
5.2	État de l'art autour des niveaux de marnage . . . . .	71
5.2.1	Marnages fixes . . . . .	71
5.2.2	Marnages fixes avec différentiation temporelle . . . . .	72
5.2.3	Marnages variants avec différentiation temporelle . . . . .	73
5.2.4	Marnages fixes avec des classes temporelles supplémentaires . . . . .	74
5.3	Contexte et motivations . . . . .	75
5.3.1	Contexte et cadre du problème . . . . .	75
5.3.2	Anatomie et points de passage du niveau d'eau de la solution optimale par RFLATS . . . . .	76
5.3.3	Complexité de RFLATS . . . . .	77
5.4	Formulation du problème de pompage . . . . .	81
5.4.1	Contraintes du réseau d'eau . . . . .	81
5.4.2	Contraintes de marnage fixe . . . . .	83
5.4.3	Contraintes correspondants aux périodes artificielles . . . . .	84
5.5	Résultats expérimentaux . . . . .	87
5.5.1	Instance de petit SDE rural . . . . .	87
5.5.2	Point de passage et robustesse de RFLATS . . . . .	88
5.5.3	Journée type : jour de semaine . . . . .	91
5.5.4	Journée type : dimanche . . . . .	95
5.5.5	Marnage reconstruit . . . . .	96
5.6	Résumé du chapitre . . . . .	99

---

Nous abordons dans ce chapitre le problème de la minimisation du coût énergétique d'un grand nombre de petits systèmes de distribution d'eau d'un grand groupe français du secteur de l'eau. Chaque système répond aux demandes par l'eau stockée dans son réservoir surélevé, qui est rempli durant des opérations de pompage programmées. Le problème d'optimisation des opérations de pompage, connu sous le nom de *Pump Scheduling Problem*, est applicable grâce à un contrôleur avancé avec une configuration centrale. Les petits réseaux ruraux, ayant peu de ressource, utilisent souvent une configuration locale avec une stratégie 'automatisée' en utilisant des niveaux de marnages, dans lesquels les pompes se mettent automatiquement en marche ou à l'arrêt dès lors que le niveau d'eau atteint une valeur donnée dans le réservoir. Nous proposons des modèles de programmation linéaire en nombres entiers pour ces différents types de marnages.

La stratégie de marnage la plus évoluée actuellement est le Reduced Fixed trigger Levels in Additional Time Slots (RFLATS) de Quintiliani, C., et Creaco, E. (2019) [88]. Dans cette thèse, nous proposons les premières approches de résolution exacte pour la gestion d'un marnage de type RFLATS en montrant sa complexité polynomiale avec la programmation dynamique et en proposant son premier modèle de programmation linéaire en nombres entiers. Nous montrons aussi que le RFLATS guide le niveau d'eau vers des points de passage du niveau d'eau, offrant une robustesse et permettent de découper l'horizon en journée type afin de formuler des sous-problèmes plus petits. La solution finale est reconstruite à partir des solutions obtenues pour les différents problèmes de journée type prises séparément. Les résultats numériques révèlent que le RFLATS apporte un meilleur coût énergétique comparé aux marnages actuellement utilisés dans les réseaux, choisis pour leur très forte résistance aux aléas de consommation.

## 5.1 Introduction

Nous abordons le problème de la minimisation du coût énergétique généré par les opérations de pompage pour alimenter les châteaux d'eau surélevés, satisfaisant les demandes des consommateurs, à partir des sources d'eau potable dans des systèmes de distribution d'eau (SDE). Les SDE sont composés d'éléments actifs (pompes, vannes...) et d'éléments passifs (sources, châteaux d'eaux/réservoirs,

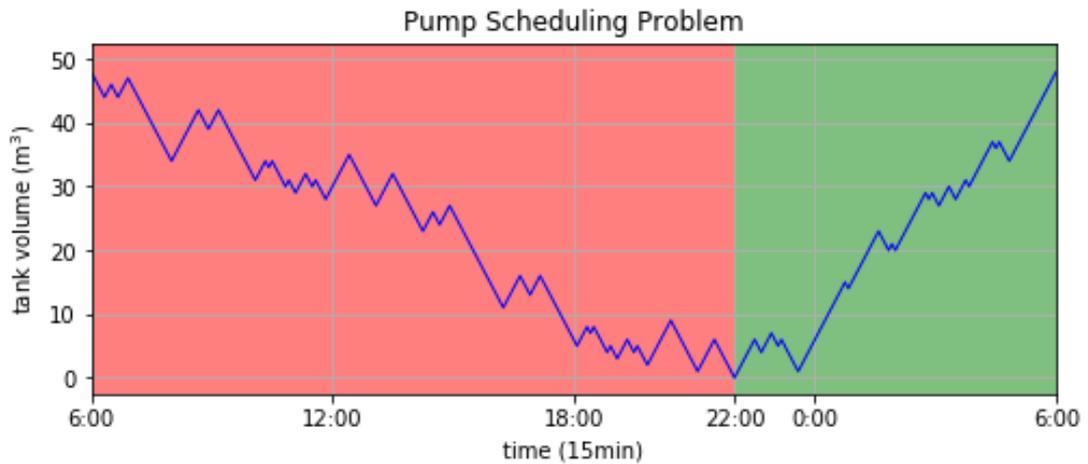


FIGURE 5.1 – Profil du niveau d'eau avec PSP

canalisations...), où l'eau est collectée à partir des sources, pressurisée dans des stations de pompage, élevée vers des châteaux d'eau par des canalisations et des vannes, et finalement fournie par gravité aux clients locaux [26]. Dans la plupart des cas, un SDE gère un groupe de plusieurs communes proches, où la topologie du réseau est choisie en fonction des contraintes physiques locales et peut être en cascade, en étoile, en arbre ou en maille. Les petits réseaux privilégieront une topologie en étoile ou en arbre du fait qu'il n'y a souvent qu'une seule source, tandis que les grandes villes préféreront une topologie maillée afin de profiter de plusieurs sources.

L'objectif primaire de la gestion d'un SDE est de minimiser le coût énergétique tout en satisfaisant les demandes des consommateurs. Ce coût énergétique, lié à l'utilisation des pompes et des vannes, est principalement composé du coût de l'électricité consommée pour les opérations de pompage. Grâce à leurs capacités de stockage, les SDE sont des instruments remarquables pour mettre en œuvre des transferts de charge, il est ainsi possible d'anticiper la demande en pompant une quantité d'eau adéquate lorsque l'électricité est la moins chère. Une solution peut être représentée par une planification de l'utilisation des pompes et des vannes du problème appelé *Pump Scheduling Problem* (PSP), largement étudiée pour la gestion des SDE. Un exemple de l'évolution du niveau d'eau d'un château d'eau est montrée dans la Figure 5.1, où on observe que le niveau d'eau baisse progressivement durant les heures pleines (zone en rouge) et monte progressivement durant les heures creuses (zone en vert).

L'objectif des PSP peut aussi prendre en compte la minimisation des émissions de gaz [20], la maximisation de résilience [45]. Les pompes peuvent être considéré comme un groupe afin de réduire le nombre de variables [48, 95, 103]. Les auteurs de [21] ont travaillé sur le choix et l'installation des pompes pour un design robuste de la station de pompage en décomposant l'horizon en journée type de consommation. D'autres travaux sont portés sur le placement des vannes unidirectionnelles [37], des vannes d'ouverture/fermeture [87], et des vannes de réduction de pression [37, 44, 87]. Des études ont également été menées sur des SDE ne comportant que de petits réservoirs d'eau [54]. Le PSP couplé à la prévision de la consommation d'eau a également été étudié et permet des économies d'énergie importantes [6, 7]. Des approches par programmation stochastique dynamique [27], par algorithme génétique avec poids adaptatif [1] et par programmation mathématique [22, 36] ont été étudiées pour le PSP. Des travaux ont également été menés sur une version en ligne du problème [64].

On retrouve des travaux sur la minimisation du nombre de changements d'état des pompes (*Number of Pumps Switches*) dans une fenêtre de temps [73], la minimisation du nombre de changements d'état de la pompe en tant que second objectif [4, 14], ou la maximisation de l'intervalle de temps entre deux changements d'état de la pompe (*Pump Switch Time Gap*) [56].

Il existe principalement deux schémas de contrôle dans les SDE, et la caractéristique principale dépend de la localisation du contrôleur [56]. Le premier schéma utilise une configuration centrale, où le contrôleur unique est installé au centre de contrôle principal i.e. à distance des réservoirs et des stations de pompage. Le deuxième schéma utilise une configuration locale, où les contrôleurs sont installés localement i.e. au niveau de chacun des réservoirs et/ou chacun des stations de pompage. La configuration centrale a l'avantage de détenir de l'information de l'entièreté du réseau, mais a l'inconvénient de la nécessité d'un effort de calcul intense afin de résoudre des problèmes d'optimisation en ligne complexes. La configuration locale est plus simple et plus robuste car les informations utiles sont directement accessibles, les problèmes associés sont plus simples et sont souvent hors ligne.

Les SDE considérés dans cette thèse sont des petits SDE ruraux, et réduit à la composition d'une station de pompage et un réservoir, dans lesquels l'eau potable est pompée directement depuis la source d'eau potable vers le réservoir

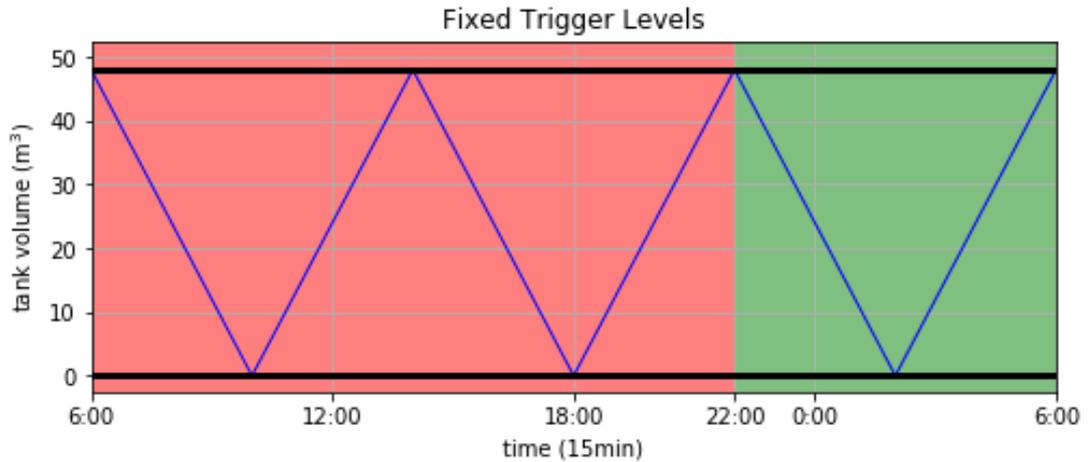


FIGURE 5.2 – Profil de marnage par FTL

final. Les pompes de l'unique station de pompage sont à débit fixe, i.e. ils délivrent une quantité fixe par unité de temps. Dans cette configuration, les petits SDE ruraux favorisent l'utilisation d'une stratégie simple et automatisée, comme celui des *Trigger Levels*, dites "niveaux de marnages". Les niveaux de marnages sont peu étudiés dans la littérature, c'est un mécanisme où les pompes se mettent automatiquement en marche ou à l'arrêt lorsque le niveau d'eau atteint une valeur donnée dans le réservoir, ainsi seulement quelques variables de décision sont nécessaires pour décrire la gestion du réservoir.

## 5.2 État de l'art autour des niveaux de marnage

### 5.2.1 Marnages fixes

L'application la plus simple des niveaux de marnage, appelée *Fixed Trigger Levels* (FTL) [67, 85], choisit deux niveaux de marnage par pompe : le niveau de marnage contrôlant le déclenchement de la pompe (appelé *marnage bas*) ; et le niveau de marnage contrôlant l'arrêt de la pompe (appelé *marnage haut*). Lorsque le niveau d'eau atteint la valeur du marnage bas, la pompe s'allumera et restera active jusqu'à ce que le niveau d'eau atteindra la valeur du marnage haut, là où la pompe s'arrêtera. Un exemple du FTL est montré dans la Figure 5.2, où on observe que le niveau d'eau (en bleu) jongle entre le marnage bas et le marnage haut (en noirs).

Les valeurs des marnages sont en pratique comparées avec le niveau d'eau donc

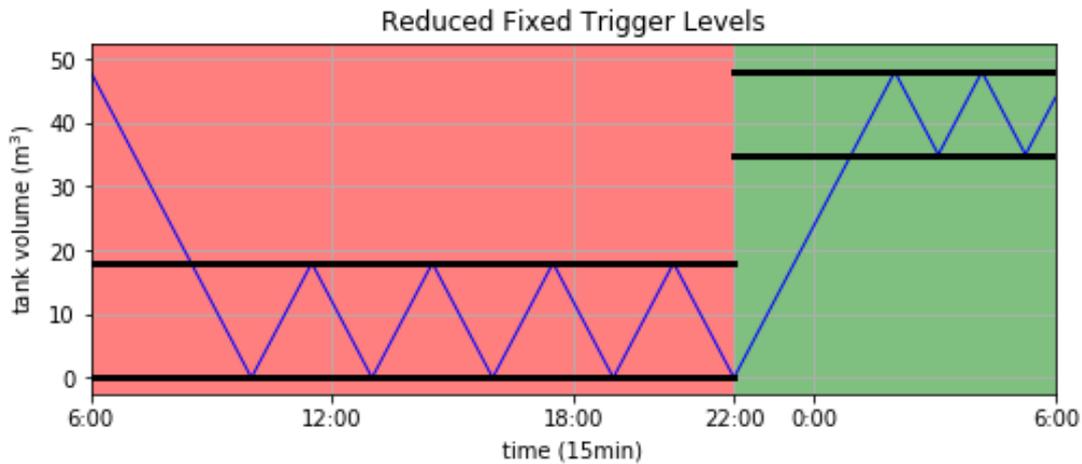


FIGURE 5.3 – Profil de marnage par RFTL

exprimées en mètres. Afin de ne pas alourdir les notation lors des comparaisons avec le volume d'eau dans le réservoir ou le débit de la pompe, on considère par la suite que les marnages sont convertis et exprimés en mètres cubes.

Souvent, le gestionnaire du réseau souhaite garantir une autonomie minimum (en heure) au cas où toutes les pompes tombent en panne en même temps, ainsi le marnage bas peut être fixé à l'équivalent en mètre cube de cette autonomie. Le marnage haut, lui est souvent fixé à la capacité maximale du réservoir. Finalement, dans ce cas précis les deux marnages sont prédéfinis et il n'y a aucune variable de décision.

Le FTL est une stratégie simple, mais il possède un inconvénient majeur : il ne permet pas de faire des économies.

## 5.2.2 Marnages fixes avec différentiation temporelle

Une première amélioration naturelle qu'on souhaite apporter à FTL est d'avoir un mécanisme "déplaçant" des consommations électriques de la catégorie temporelle heures pleines (noté **P**) vers la catégorie temporelle heures creuses (noté **C**). Plus précisément, on cherche à avoir un réservoir d'eau rempli au début de **P** et d'essayer de le vider totalement à la fin de **P**, ainsi une grosse partie de l'électricité est alors utilisée durant **C** pour le remplissage. Ce comportement espéré peut être obtenu par exemple si les deux marnages en **P** sont très bas et si les deux marnages en **C** sont très haut.

L'approche appelée Reduced Fixed Trigger Levels (RFTL) de Marchi, A., et

al. (2016) [75], définit un couple de marnage haut et bas par catégorie temporelle (i.e.  $\mathbf{P}$  ou  $\mathbf{C}$ ). Un exemple du RFTL est montré dans la Figure 5.3, où on observe un couple de marnage de faible valeur en  $\mathbf{P}$  et un couple de plus grande valeur en  $\mathbf{C}$ , et menant globalement à un vidage en  $\mathbf{P}$  et un remplissage en  $\mathbf{C}$ .

Comme pour FTL, dans la plupart des cas le marnage bas en  $\mathbf{P}$  peut être fixé au volume correspondant à l'autonomie minimum garantie et le marnage haut en  $\mathbf{C}$  est fixé à la capacité maximale du réservoir. Finalement, ce problème a seulement deux variables de décision : marnage haut en  $\mathbf{P}$  et marnage bas de  $\mathbf{C}$ .

Dans la suite, on dénote par *marnage serré* un marnage où la différence entre le marnage haut et le marnage bas est faible (e.g. inférieur à  $\frac{1}{3}$  de la capacité du réservoir), et inversement par *marnage lâche* lorsque cette différence est grande (e.g. supérieur à  $\frac{2}{3}$  de la capacité du réservoir). On utilise aussi la notion de *marnage neutre* pour désigner un marnage non-lâche et non-serré (e.g. environ à  $\frac{1}{2}$  de la capacité du réservoir).

Une stratégie de type RFTL est facilement programmable sur les automates gérants des niveaux de marnages, il suffit de définir les deux plages horaires correspondant aux heures pleines et aux heures creuses. Le RFTL génère des économies maximales en appliquant un marnage serré de faible valeur durant les heures pleines et un marnage serré de grande valeur durant les heures creuses. Cette gestion a l'inconvénient de générer un trop grand nombre de changements d'état des pompes, et réduit fortement la durée de vie des pompes.

### 5.2.3 Marnages variants avec différentiation temporelle

L'idée du time-Variable Trigger Levels (VTL) de Alvisi, S., et Franchini, M. (2017) [5] est, qu'au début de  $\mathbf{P}$  ou de  $\mathbf{C}$  on commence avec un certain marnage puis les marnages se rapprochent avec le temps. Un exemple du VTL est montré dans la Figure 5.4, où les deux marnages intéressants varient dans le temps, ici le marnage haut de  $\mathbf{P}$  décroît linéairement et le marnage bas de  $\mathbf{C}$  croît linéairement.

Un programme mathématique est présenté dans Housh, M., et Salomons, E. (2019) [56] où VTL utilise des fonctions linéaires pour les deux marnages intéressants. Comme pour FTL ou RFTL, deux des marnages peuvent être fixés, dans ce cas les variables de décisions sont entièrement définies par les écarts entre le marnage haut et le marnage bas, à la fin de  $\mathbf{P}$  et à la fin de  $\mathbf{C}$ .

Le VTL produit un nombre de changements d'état des pompes largement plus faible que pour RFTL, mais ce nombre est souvent assez grand durant les pas de

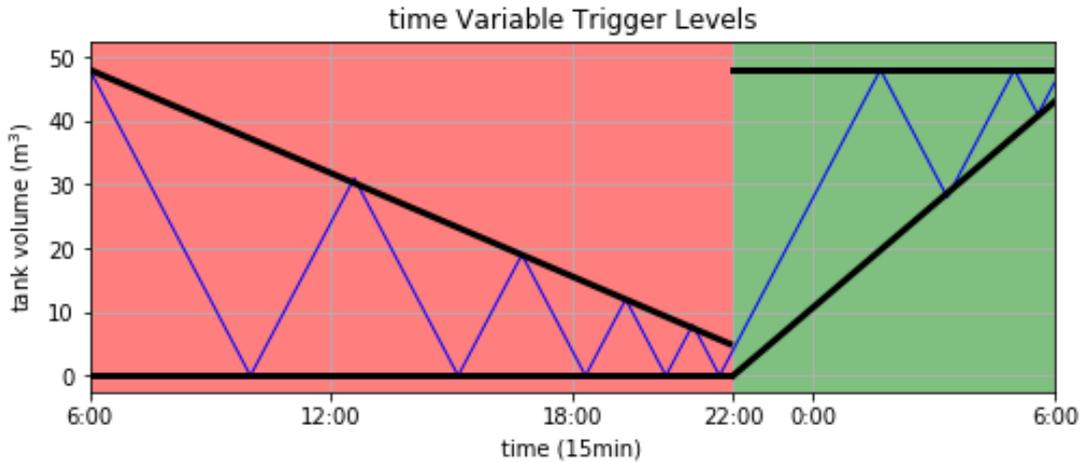


FIGURE 5.4 – Profil de marnage par VTL

temps proches des passages de heures pleines à heures creuses et vice-versa.

#### 5.2.4 Marnages fixes avec des classes temporelles supplémentaires

Le Reduced Fixed trigger Levels in Additional Time Slots (RFLATS) de Quintiliani, C., et Creaco, E. (2019) [88] utilise une configuration similaire à RFTL, et subdivise les catégories génériques ( $\mathbf{P}$  et  $\mathbf{C}$ ) en considérant une nouvelle catégorie artificielle ( $\mathbf{P}'$  et  $\mathbf{C}'$ ). Plus précisément, la catégorie temporelle 'heures pleines' (resp. 'heures creuses') est ici composée de  $\mathbf{P}$  puis de  $\mathbf{P}'$  (resp. de  $\mathbf{C}$  puis de  $\mathbf{C}'$ ). Un couple de marnage est affecté pour chacune des quatre catégories (génériques ou artificielles). Un exemple du RFLATS est montré dans la Figure 5.5, où on observe deux couples de marnages au sein des heures pleines et deux couples de marnages au sein des heures creuses, chacun des couples contrôle le niveau d'eau de manière différente. On y observe aussi que le marnage de 19h00 à 22h00 permet d'arrêter "brutalement" la pompe à 19h00, le moment où les heures pleines artificielles commencent.

De manière générale, pour RFLATS on souhaite que  $\mathbf{P}$  et  $\mathbf{C}$  utilisent des couples de marnage pour minimiser le nombre de changements d'état de la pompe, puis que  $\mathbf{P}'$  et  $\mathbf{C}'$  utilise des couples de marnage pour guider le niveau d'eau au plus bas ou au plus haut du réservoir afin de faire un bénéfice sur l'électricité. Ainsi on peut s'imaginer que  $\mathbf{P}$  et  $\mathbf{C}$  utilisent des marnages lâches, où par exemple on fixe leurs marnages bas à l'équivalent en mètre cube de l'autonomie minimum

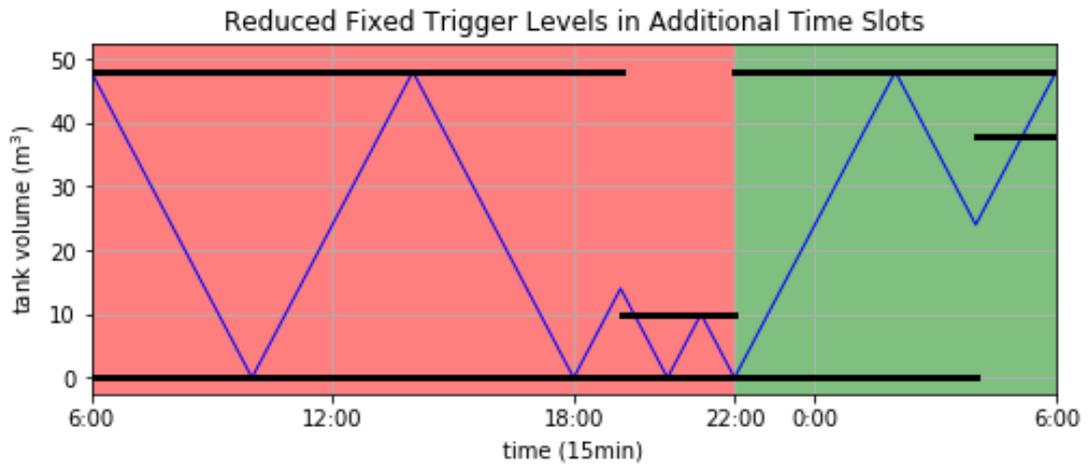


FIGURE 5.5 – Profil de marnage par RFLATS

garantie et on fixe leurs marnages hauts à la capacité maximale du réservoir. On peut aussi s’imaginer que  $P'$  et  $C'$  utilise des marnages serrés. Finalement, le problème RFLATS peut se résumer à la recherche sur quatre variables de décision : deux variables pour le marnage haut durant  $P'$  et pour le marnage bas durant  $C'$ , et deux variables pour la durée de  $P'$  et  $C'$ .

## 5.3 Contexte et motivations

### 5.3.1 Contexte et cadre du problème

Les travaux autour PSP utilisent une configuration centrale, tandis que les travaux autour des marnages utilisent une configuration locale et sont focalisés sur un seul réservoir, de plus ils ignorent les questions autour de la pression (e.g. la loi de Darcy-Weisbach ou la pression de refoulement ...). Dans cette thèse, pour chaque semaine, on se place dans un petit SDE rural possédant qu’une seule station de pompage et qu’un seul château d’eau délivrant toute la zone, dans lequel on cherche à minimiser le coût des opérations de pompage tel que la solution produit un nombre de changements d’état des pompes acceptable. Le SDE est muni de capteurs pour le marnage de type RFLATS, et qui sont programmés pour faire une action d’activation ou arrêt d’une pompe dès que le niveau d’eau dans le château d’eau atteint une certaine valeur durant des plages horaires prédéfinies. La station de pompage est équipée de deux ou plusieurs

pompes, où seule la pompe principale peut-être active, les autres étant des back-ups. Le statut principal ou back-up est échangé tous les quelques jours afin que les pompes vieillissent en même temps.

Du fait de la discrétisation du temps, on considère que le remplissage (i.e. le pompage) et la consommation (i.e. la demande) se font en même temps à la fin de chaque pas de temps, ainsi le remplissage peut directement être utilisé pour répondre à la demande. De plus les pompes considérées sont à débit fixe, ainsi le remplissage sera de 0 ou égale à la quantité maximale délivrable par la pompe durant chaque pas de temps.

En France, la plupart des contrats d'électricité, intégrant la différenciation temporelle, propose des heures pleines de 6h00 à 21h59 et des heures creuses de 22h00 à 5h59. Notons que les jours fériés et les dimanches sont entièrement des heures creuses. L'étude est menée pour chaque semaine de Lundi 6h00 au Lundi 5h59 suivant. À la fin de chaque semaine, de Dimanche 22h00 à Lundi 5h59, on se donne 8 heures maximum pour définir les nouveaux niveaux de marnages de la semaine suivante. L'objectif est ainsi de trouver une solution exacte permettant de réaliser le maximum d'économie sur les opérations de pompage avec RFLATS.

### 5.3.2 Anatomie et points de passage du niveau d'eau de la solution optimale par RFLATS

En général, afin de générer le plus d'économie, le marnage de type RFLATS souhaite que  $\mathbf{P}$  et  $\mathbf{C}$  utilisent des couples de marnage lâche, puis que  $\mathbf{P}'$  et  $\mathbf{C}'$  utilisent des couples de marnage serré. Dans la [Figure 5.6](#), on peut observer trois marnages de types RFLATS ayant à peu près le même coût. Dans l'ordre, le premier marnage est un marnage neutre, son problème est qu'il ne contrôle assez peu le niveau d'eau et autorisera de grand écart en présence d'incertitudes de la demande. Le deuxième marnage est un marnage très serré, son problème est qu'il produit un très grand nombre de changements d'état de la pompe, ce genre de solution n'est pas accepté dans cette thèse (on va forcer un écart minimum entre le marnage haut et le marnage bas). Le troisième marnage est le type de solution qu'on recherche, i.e. un marnage lâche sur une longue durée accompagné d'un marnage serré sur une courte période; idéalement il n'y a qu'un ou deux changement d'état durant cette courte période. Ce type de solution va donc chercher à vider au maximum le réservoir à la fin de chaque  $\mathbf{P}$  et à remplir le réservoir au

maximum à la fin de chaque  $\mathbf{C}$ .

Le RFTL, qui est actuellement répandu dans les petits SDE ruraux, existe principalement sous la forme de marnage neutre. Ce type de marnage est peu résistant aux incertitudes de la demande, en effet si la demande réelle dévie légèrement par rapport à l'instance prédite utilisée pour l'optimisation hors ligne, il est ainsi possible que durant l'application réelle qu'un marnage haut ou qu'un marnage bas ne se déclenche pas au moment prévu. La conséquence est que le niveau d'eau peut être complètement désynchronisé par rapport à ce qui était prévu durant le reste de la période d'étude. Ainsi, on souhaite généralement un meilleur contrôle du niveau d'eau en utilisant un marnage serré. Dans cette thèse, on a ainsi remarqué que le marnage de type RFLATS offre une meilleure robustesse pour la gestion du réseau. En effet, si une désynchronisation se produit, alors elle semble pouvoir être facilement corrigée par le marnage serré.

On découvre que l'avantage de RFLATS, nous permet de découper l'horizon d'une semaine en petits morceaux et de se ramener à la résolution de problèmes plus petits. Par la suite, la période d'étude totale d'une semaine sera découpée en journée type de 24 heures (avec un pas de temps de 15 minutes). Ces journées types s'étendent de 6h00 à 5h59 du lendemain, afin de capturer complètement les catégories temporelles (e.g. la journée type 'jour de semaine' peut s'étendre de lundi 6h00 à mardi 5h59). Les journées types sont donc composées d'un  $\mathbf{P}$  puis d'un  $\mathbf{C}$  (pour les jours de semaine, samedi ...) ou seulement d'un unique  $\mathbf{C}$  (pour les dimanches, jours fériés ...).

### 5.3.3 Complexité de RFLATS

Dans la littérature, le problème RFLATS est uniquement résolu par une énumération totale (pour le cas ayant qu'une seule pompe) ou par l'algorithme évolutionniste *Borg Multiobjective Evolutionary Algorithm* (BorgMOEA) [88]. Le BorgMOEA n'est pas un algorithme unique, il représente plutôt une classe d'algorithmes dont les opérateurs sont sélectionnés de manière adaptative en fonction du problème, et il combine  $\epsilon$ -dominance, une mesure de la vitesse de convergence appelée  $\epsilon$ -progress, des redémarrages aléatoires et une recombinaison multi-opérateurs auto-adaptative dans un cadre d'optimisation unifié [51]. À titre indicatif, si l'évaluation d'une certaine solution RFLATS prend  $0.2ms$ , que les 2 marnages peuvent prendre 1000 valeurs différentes et que les durées de  $\mathbf{P}'$  et  $\mathbf{C}'$  peuvent prendre respectivement 32 et 16 valeurs possibles, il faut alors

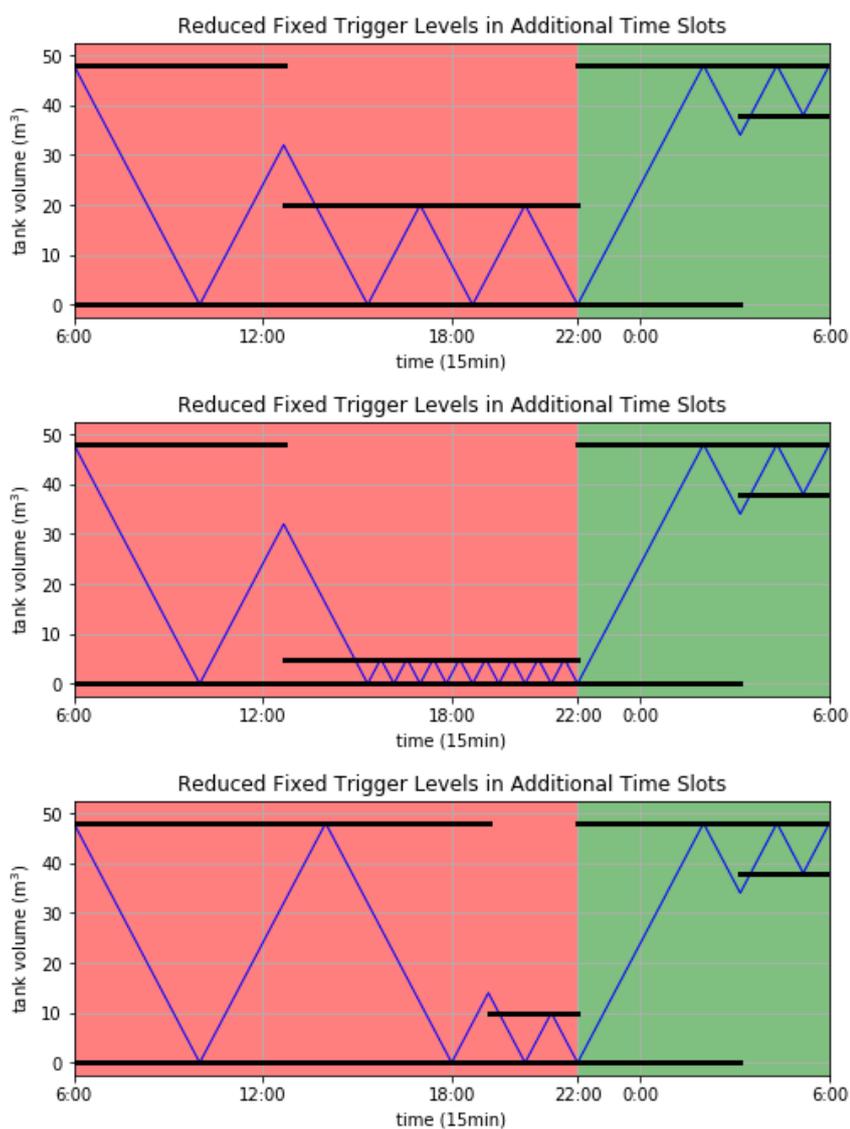


FIGURE 5.6 – Différentes solutions de marnage de type RFLATS

28 heures pour trouver la solution optimale par énumération complète, ce qui dépasse largement les 8 heures qu'on s'autorise.

Dans cette thèse, on souhaite trouver des solutions exactes pour la résolution de RFLATS. Dans un premier temps, on propose un algorithme de programmation dynamique, où on recherche le plus court chemin dans un graphe séquentiel représentant l'évolution du niveau d'eau pour toutes les solutions de l'espace de recherche. Chaque sommet créé représente un moment où il y a changement d'état de la pompe pour certain un ensemble de solution (i.e. des marnages de type RFLATS) ; et chaque chemin, i.e. une série de sommet, représente l'évolution du niveau d'eau pour un certain ensemble de solution.

On modélise les sommets avec les informations suivantes : un pas de temps  $t$ , un niveau d'eau  $v$  et un ensemble de solutions  $E$ . Remarque, pour simplifier les affichages, les sommets peuvent être placés aux coordonnées  $(t, v)$  dans une grille, comportant donc uniquement l'information de  $E$ . Les transitions sortantes d'un sommet  $(t, v, E)$  correspondent à une partition  $\{E_{t_1}, \dots, E_{t_n}\}$  de  $E$ , où les  $n$  sommets terminaux ont un pas de temps  $t_1, \dots, t_n$  différent et où chacun de ces sommets correspond à un changement d'état de la pompe au pas de temps  $t_i$  pour le sous-ensemble  $E_{t_i}$  (pour  $1 \leq i \leq n$ ). Notons que les niveaux d'eau  $v_i$  sont déduit à partir de l'état de la pompe et de la demande entre  $t_1$  et  $t_i$ . Lorsqu'une transition correspond à celle d'un remplissage du réservoir, alors son coût de pompage est calculé en se basant sur le prix de l'électricité entre  $t$  et  $t_i$ .

La Figure 5.7 illustre une partie des sommets et transitions d'un tel graphe durant les heures pleines, les ensembles de solution sont ainsi représentés par des couples (début de  $\mathbf{P}'$  et son marnage haut). Dans cette figure, le symbole  $\forall$  signifie "tout l'ensemble de définition", et la notation " $>$ " (resp. " $<$ ") signifie "les valeurs de l'ensemble de définition supérieur à (resp. inférieur à)". Comme pour les précédentes figures, les évolutions du niveau d'eau (i.e. les transitions du graphe) sont tracées en bleu. On observe dans cette figure la partition du sommet  $(\forall, \forall)$  placé au coordonnées  $(10, 0)$ , où les transition mènent vers des sommets aux coordonnées  $(18, 200)$ ,  $(27, 450)$  et  $(40, 750)$  placés sur une ligne droite.

Soient  $T$  le nombre de pas de temps,  $\underline{V}$  (resp.  $\bar{V}$ ) la capacité minimale (resp. maximale) du réservoir et  $Q$  le débit maximal de la pompe sur un pas de temps en  $m^3$ . La demande est déterministe, ainsi de  $t = 0$  à  $t = t'$  la demande totale est une quantité fixe. Lorsque la pompe délivre uniquement 0 ou  $Q$  sur chaque pas de temps, alors les volumes d'eau possible dans le réservoir en  $t'$  dépendent

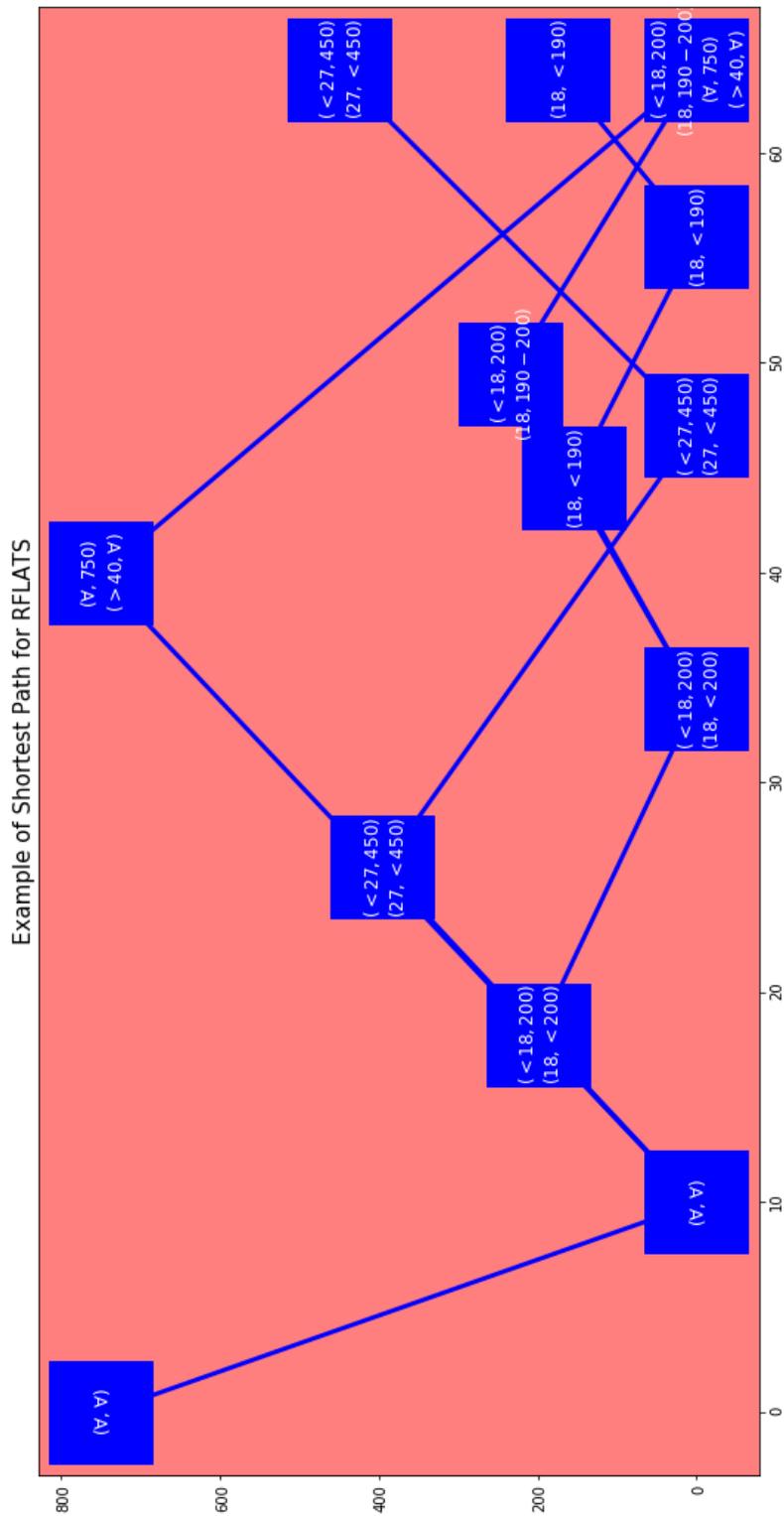


FIGURE 5.7 – Plus court chemin pour le marnage optimal durant les heures pleines

uniquement du nombre de pas de temps où la pompe a été active. Ainsi, pour un certain pas de temps  $t'$ , les sommets ont un écart correspondant  $n * Q$  entre eux (avec  $n$  un entier non nul). Soit  $N = \frac{\bar{V}-V}{Q}$ , le nombre maximal de sommet présents pour un certain pas de temps est de  $N$ , ainsi le nombre maximum de sommets de ce graphe est au nombre de  $O(T * N)$ . De même, le nombre maximal de transitions sortantes d'un sommet est de  $\lceil N \rceil$ , ainsi le nombre maximum de transitions de ce graphe est au nombre de  $O(T * N^2)$ . Finalement, la complexité polynomiale de RFLATS est montrée en cherchant le plus court chemin dans ce graphe.

À titre indicatif, dans l'instance utilisée dans la [Section 5.5](#), le graphe en question aurait au maximum 1152 sommets et 13824 transitions. Cependant, la création de ce graphe nécessite de bonnes compétences en développement et en modélisation, afin de trouver les partitions de l'ensemble de solutions de chaque sommet. Dans la suite, nous proposons un modèle plus souple utilisant la programmation linéaire en nombres entiers. Ce modèle pourra par exemple être adapté plus facilement pour l'utilisation de plusieurs pompes ou pour intégrer des déviations autour de la consommation.

## 5.4 Formulation du problème de pompage

Dans cette section, nous proposons des modèles de programmation linéaire en nombres entiers pour les différents problèmes (PSP, FTL, RFTL et RFLATS). Nous classons les contraintes en différents types : contraintes du réseau d'eau, contraintes de marnage, contraintes correspondant aux périodes artificielles de RFLATS.

### 5.4.1 Contraintes du réseau d'eau

Les contraintes de réseau d'eau décrivent le fonctionnement générale du réseau, elles sont communes à tous nos problèmes.

Soient  $x_t$  le booléen d'état actif/éteint de la pompe durant  $t \in T$ ,  $q_t$  le débit de la pompe durant  $t \in T$ . Dans [\[22\]](#), la puissance électrique consommée (en kWh) est approximée par une fonction linéaire du débit réel  $q_t$  et l'état booléen  $x_t$ . Soit  $Q$  le débit maximal de la pompe sur un pas de temps en  $m^3$ . Dans le cas de pompes à débit fixe,  $q_t$  ne peut prendre que la valeur 0 ou  $Q$ , ainsi on modélise

la puissance électrique consommée par :

$$PC(q_t) = P * q_t \quad (5.1)$$

Avec  $P$  un coefficient caractérisant la puissance de la pompe (en kWh/m<sup>3</sup>).

Soit  $v_t$  le volume d'eau dans le château d'eau au début de  $t \in T$  et borné par la capacité minimale  $\underline{V}$  et maximale  $\bar{V}$  du réservoir. On dénote par  $C_t$ , le prix de l'électricité durant  $t$  (en €/kWh), suivant le contact d'électricité, ce prix est prédéterminé ou dépend du prix réelle à l'instant de la consommation. Comme pour les modèles de marnage dans la littérature, nous ne prenons pas en compte les contrainte de charge hydraulique. La gestion de type PSP d'un château d'eau peut être formulée comme suit :

$$\min \sum_{t \in T} C_t * PC(q_t) + C_{\mathbf{P}} * PC(\bar{V} - v_T) \quad (5.2)$$

$$\text{s.t. } v_{t+1} = v_t + q_t - D_t, \quad \forall t \quad (5.3)$$

$$v_0 = \bar{V} \quad (5.4)$$

$$q_t = Q * x_t, \quad \forall t \quad (5.5)$$

$$x_t \in \{0, 1\}, 0 \leq q_t, v_t \in [\underline{V}, \bar{V}]$$

L'objectif [Équation \(5.2\)](#) minimise le coût énergétique du pompage, et si le réservoir n'est pas rempli à la fin de la période, alors on ajoute le coût d'une pénalité équivalent à un remplissage complet du réservoir au prix des heures pleines  $C_{\mathbf{P}}$  (prix moyen ou prix maximal de  $\mathbf{P}$ ). La conservation de flot au niveau du château d'eau est assurée par [Équation \(5.3\)](#), où  $D_t$  est la demande à la fin de  $t$  (en m<sup>3</sup>). Le niveau d'eau à 6h00 est fixé à la capacité du réservoir [Équation \(5.4\)](#). Le débit fourni est garanti par [Équation \(5.5\)](#).

**Pénalité du réservoir non rempli** On ne force pas à ce que le réservoir soit totalement rempli à la fin de la période d'étude, car cette contrainte est trop stricte lorsqu'on utilise des marnages et que la pompe ne peut délivrer que 0 ou  $Q$ . À la place, on pénalise le manque  $\bar{V} - v_T$  par le prix moyen (ou prix maximal) durant  $\mathbf{P}$  afin d'obtenir une fonction objectif cohérente entre prix du pompage et prix de la pénalité (cf. [Équation \(5.2\)](#)).

**Modèle pour PSP simplifié** Pour chaque journée type, ce problème PSP simplifié (5.2-5.5) est un petit problème de flot et peut être résolu facilement e.g. par programmation dynamique.

### 5.4.2 Contraintes de marnage fixe

Les contraintes de marnage décrivent le fonctionnement des marnages. Soient  $\underline{v}_t$  et  $\bar{v}_t$  le marnage bas et le marnage haut durant  $t \in T$ , et bornés par la capacité minimale  $\underline{V}$  et maximale  $\bar{V}$  du réservoir (rappel : en volume, pas en hauteur d'eau). Dans le cas où la pompe ne changera pas d'état, les booléens  $\underline{a}_t$  et  $\bar{a}_t$  indiquent si le niveau d'eau risque d'être inférieur ou supérieur aux marnages en question à la fin de  $t$ , ces variables indiquent donc s'il faut changer d'état de la pompe au pas de temps  $t$ . Ainsi, l'état de la pompe en  $t$  est garanti par :

$$-\bar{a}_t * \bar{V} \leq \bar{v}_t - (v_t + q_{t-1} - D_t) \leq (1 - \bar{a}_t) * \bar{V}, \forall t \quad (5.6)$$

$$-\underline{a}_t * \underline{V} \leq -\underline{v}_t + (v_t + q_{t-1} - D_t) \leq (1 - \underline{a}_t) * \underline{V}, \forall t \quad (5.7)$$

$$x_{t-1} + \underline{a}_t - \bar{a}_t \leq 2x_t \leq x_{t-1} + \underline{a}_t - \bar{a}_t + 1, \quad \forall t \quad (5.8)$$

$$\bar{a}_t, \underline{a}_t \in \{0, 1\}, \bar{v}_t, \underline{v}_t \in [\underline{V}, \bar{V}]$$

Où à chaque pas de temps, on compare les marnages avec le niveau d'eau à la fin de  $t$  i.e. en regardant  $v_t + q_{t-1} - D_t$ . Si  $\underline{a}_t = 0$  et  $\bar{a}_t = 0$  alors la pompe garde son ancien état, sinon la pompe est forcée sur l'état actif ou éteint avec [Équation \(5.8\)](#). Notons que la formulation de l'[Équation \(5.8\)](#) permet à  $\underline{a}_t$  ou  $\bar{a}_t$  de valoir 1 durant plusieurs pas de temps à la suite, et c'est ce qu'on souhaite observer durant le passage de **P** à **C** où en général le marnage bas passent d'une petite valeur à une grande valeur, de cette manière  $v_t$  peut être inférieur à  $\underline{v}_t$  durant plusieurs pas de temps. C'est pour cette raison qu'une formulation de type  $x_t = x_{t-1} + \underline{a}_t - \bar{a}_t$  n'est pas correcte.

On introduit aussi un équivalent des marnages des pas de temps  $t$  pour chaque catégorie  $k \in K$ . Soient  $\underline{v}_k$  et  $\bar{v}_k$  le marnage bas et le marnage haut d'une catégorie  $k = \mathbf{P}$  ou  $\mathbf{C}$ , et bornés par la capacité minimale  $\underline{V}$  et maximale  $\bar{V}$  du réservoir (rappel : en volume, pas en hauteur d'eau). On dénote ces marnages par  $\underline{v}_{\mathbf{P}}, \bar{v}_{\mathbf{P}}, \underline{v}_{\mathbf{C}}$  et  $\bar{v}_{\mathbf{C}}$ , les relations entre le temps  $t \in T$  et la catégorie sont garanties par :

$$\bar{v}_t = \bar{v}_{\mathbf{C}}, \forall t \in T[\mathbf{C}] \quad (5.9)$$

$$\underline{v}_t = \underline{v}_{\mathbf{C}}, \forall t \in T[\mathbf{C}] \quad (5.10)$$

$$\bar{v}_t = \bar{v}_{\mathbf{P}}, \forall t \in T[\mathbf{P}] \quad (5.11)$$

$$v_t = v_{\mathbf{P}}, \forall t \in T[\mathbf{P}] \quad (5.12)$$

$$\bar{v}_{\mathbf{C}}, v_{\mathbf{C}}, \bar{v}_{\mathbf{P}}, v_{\mathbf{P}} \in [\underline{V}, \bar{V}]$$

Avec  $T[\mathbf{C}]$  (resp.  $T[\mathbf{P}]$ ) les pas de temps  $t$  qui sont en heures creuses (resp. heures pleines). Si  $t \in T[\mathbf{C}]$ , alors les marnages  $v_t$  et  $\bar{v}_t$  sont ceux des heures creuses, sinon ce sont ceux des heures pleines.

**Écart minimum entre marnages** Dans la [Sous-section 5.3.2](#), nous proposons de fixer un écart minimale entre le marnage haut et le marnage bas afin de ne pas produire un nombre de changements d'état de la pompe aberrant. Par exemple, si on souhaite que les changements d'états se passent avec un intervalle d'au moins 30 minutes, alors on peut choisir l'écart minimum en fonction du débit horaire de la pompe et le cumul maximal de demande durant 30 minutes. Dans cette approche, le nombre changement d'état de la pompe n'est pas lié a des variables de décision, ainsi on n'a pas besoin d'introduire des objectifs multi-critères ou d'introduire des poids entre le coût énergétique du pompage et le nombre de changements d'état, et nécessitant ainsi d'introduire des variables binaires et de certaines linéarisations. Soit  $E$  l'écart minimal entre les marnages hauts et les marnages bas, et définit comme suit :

$$v_t + E \leq \bar{v}_t, \forall t \quad (5.13)$$

**Modèle pour FTL et RFTL** Notre RFTL peut donc être modélisé par ([5.2-5.13](#)), et similairement pour notre FTL sans faire la différentiation temporelle des  $\mathbf{P}$  et  $\mathbf{C}$  dans les contraintes [Équation \(5.9\)](#) à [Équation \(5.13\)](#).

### 5.4.3 Contraintes correspondants aux périodes artificielles

Les contraintes correspondant aux périodes artificielles décrivent la détermination des périodes additionnelles, elles ajoutent aussi une mise à jour de certaines contraintes de marnage.

Le RFLATS utilise les périodes génériques  $\mathbf{P}$  et  $\mathbf{C}$ , et les périodes artificielles  $\mathbf{P}'$  et  $\mathbf{C}'$ . Soit la variable  $l_{\mathbf{P}'}$  (resp.  $l_{\mathbf{C}'}$ ) définissant la longueur de  $\mathbf{P}'$  (resp.  $\mathbf{C}'$ ) en nombre de pas de temps. Soit  $e_t^{\mathbf{P}'}$  (resp.  $e_t^{\mathbf{C}'}$ ) le booléen indiquant si  $t$  fait parti de  $\mathbf{P}'$  (resp.  $\mathbf{C}'$ ). La définition des périodes artificielles respecte les contraintes

suivantes :

$$\sum_{t \in T[\mathbf{P}]} e_t^{\mathbf{P}'} = l_{\mathbf{P}'} \quad (5.14)$$

$$\sum_{t \in T[\mathbf{C}]} e_t^{\mathbf{C}'} = l_{\mathbf{C}'} \quad (5.15)$$

$$e_t^{\mathbf{P}'} \leq e_{t+1}^{\mathbf{P}'}, \forall t \in T[\mathbf{P}] \quad (5.16)$$

$$e_t^{\mathbf{C}'} \leq e_{t+1}^{\mathbf{C}'}, \forall t \in T[\mathbf{C}] \quad (5.17)$$

$$l_{\mathbf{P}'} \in \{0, \dots, 64\}, l_{\mathbf{C}'} \in \{0, \dots, 32\}, e_t^{\mathbf{P}'}, e_t^{\mathbf{C}'} \in \{0, 1\}$$

Où les contraintes [Équation \(5.14\)](#) et [Équation \(5.15\)](#) contrôlent la durée de  $\mathbf{P}'$  et  $\mathbf{C}'$ , les contraintes [Équation \(5.16\)](#) et [Équation \(5.17\)](#) gèrent l'affectation des pas de temps qui sont en  $\mathbf{P}'$  et  $\mathbf{C}'$ .

Avec les périodes artificielles, le marnage doit respecter les contraintes suivantes :

$$\bar{v}_t = (1 - e_t^{\mathbf{C}'}) * \bar{v}_{\mathbf{C}} + e_t^{\mathbf{C}'} * \bar{v}_{\mathbf{C}'}, \forall t \in T[\mathbf{C}] \quad (5.9')$$

$$\underline{v}_t = (1 - e_t^{\mathbf{C}'}) * \underline{v}_{\mathbf{C}} + e_t^{\mathbf{C}'} * \underline{v}_{\mathbf{C}'}, \forall t \in T[\mathbf{C}] \quad (5.10')$$

$$\bar{v}_t = (1 - e_t^{\mathbf{P}'}) * \bar{v}_{\mathbf{P}} + e_t^{\mathbf{P}'} * \bar{v}_{\mathbf{P}'}, \forall t \in T[\mathbf{P}] \quad (5.11')$$

$$\underline{v}_t = (1 - e_t^{\mathbf{P}'}) * \underline{v}_{\mathbf{P}} + e_t^{\mathbf{P}'} * \underline{v}_{\mathbf{P}'}, \forall t \in T[\mathbf{P}] \quad (5.12')$$

$$\bar{v}_t, \underline{v}_t, \bar{v}_{\mathbf{C}}, \underline{v}_{\mathbf{C}}, \bar{v}_{\mathbf{C}'}, \underline{v}_{\mathbf{C}'}, \bar{v}_{\mathbf{P}}, \underline{v}_{\mathbf{P}}, \bar{v}_{\mathbf{P}'}, \underline{v}_{\mathbf{P}'}, \in [\underline{V}, \bar{V}]$$

Où le marnage dépend de si  $t$  est en  $\mathbf{P}$  (resp.  $\mathbf{C}$ ) ou en  $\mathbf{P}'$  (resp.  $\mathbf{C}'$ ), déterminé par le booléen  $e_t^{\mathbf{P}'}$  (resp.  $e_t^{\mathbf{C}'}$ ).

**Linéarisation des contraintes de marnage** Les contraintes [Équation \(5.10'\)](#) et [Équation \(5.11'\)](#), qui sont des contraintes quadratiques ayant des termes de type  $p * q$ , multipliant une variable binaire  $p$  avec une variable continue  $q$  bornée par  $Q$ , peuvent facilement être linéarisable. En effet, on peut linéariser le produit  $z = p * q$  en utilisant les contraintes suivantes :

$$z = p * q \Leftrightarrow \begin{cases} z \geq 0 \\ z \leq p * Q \\ z \leq q \\ z \geq q - (1 - p) * Q \end{cases}$$

**Modèle pour RFLATS** Un RFLATS peut être modélisé par (5.2-5.8, 5.9'-5.12', 5.13-5.17). Dans notre modélisation, on utilise au maximum la capacité du réservoir, ainsi on considère que les variables  $v_C$ ,  $v_P$  et  $v_{P'}$  seront fixées à  $\underline{V}$ , ainsi que  $\bar{v}_C$ ,  $\bar{v}_P$  et  $\bar{v}_{C'}$  seront fixées à  $\bar{V}$ . Ainsi les variables de décisions du modèle sont :  $l_{P'}$ ,  $l_{C'}$ ,  $\bar{v}_{P'}$  et  $v_{C'}$ . Finalement, avec la linéarisation des termes  $z_t^{C'} = e_t^{C'} * v_{C'}$  et  $\bar{z}_t^{P'} = e_t^{P'} * \bar{v}_{P'}$  des contraintes Équation (5.10') et Équation (5.11'), le modèle entier de RFLATS peut s'écrire de la manière suivante :

$$\text{RFLATS : } \min \sum_{t \in T} C_t * PC(q_t) + C_P * PC(\bar{V} - v_T) \quad (5.2)$$

$$\text{subject to } v_{t+1} = v_t + q_t - D_t, \quad \forall t \quad (5.3)$$

$$v_0 = \bar{V} \quad (5.4)$$

$$q_t = Q * x_t, \quad \forall t \quad (5.5)$$

$$-\bar{a}_t * \bar{V} \leq \bar{v}_t - (v_t + q_{t-1} - D_t) \leq (1 - \bar{a}_t) * \bar{V}, \forall t \quad (5.6)$$

$$-\underline{a}_t * \underline{V} \leq -v_t + (v_t + q_{t-1} - D_t) \leq (1 - \underline{a}_t) * \underline{V}, \forall t \quad (5.7)$$

$$x_{t-1} + \underline{a}_t - \bar{a}_t \leq 2x_t \leq x_{t-1} + \underline{a}_t - \bar{a}_t + 1, \quad \forall t \quad (5.8)$$

$$\bar{v}_t = \bar{V}, \quad \forall t \in T[\mathbf{C}] \quad (5.9'')$$

$$v_t = (1 - e_t^{C'}) * \underline{V} + z_t^{C'}, \quad \forall t \in T[\mathbf{C}] \quad (5.10''\text{a})$$

$$v_{C'} - (1 - e_t^{C'}) * \bar{V} \leq z_t^{C'} \leq v_{C'}, \quad \forall t \in T[\mathbf{C}] \quad (5.10''\text{b})$$

$$0 \leq z_t^{C'} \leq e_t^{C'} * \bar{V}, \forall t \in T[\mathbf{C}] \quad (5.10''\text{c})$$

$$\bar{v}_t = (1 - e_t^{P'}) * \bar{V} + \bar{z}_t^{P'}, \quad \forall t \in T[\mathbf{P}] \quad (5.11''\text{a})$$

$$\bar{v}_{P'} - (1 - e_t^{P'}) * \bar{V} \leq \bar{z}_t^{P'} \leq \bar{v}_{P'}, \quad \forall t \in T[\mathbf{P}] \quad (5.11''\text{b})$$

$$0 \leq \bar{z}_t^{P'} \leq e_t^{P'} * \bar{V}, \forall t \in T[\mathbf{P}] \quad (5.11''\text{c})$$

$$v_t = \underline{V}, \quad \forall t \in T[\mathbf{P}] \quad (5.12'')$$

$$v_t + E \leq \bar{v}_t, \forall t \quad (5.13)$$

$$\sum_{t \in T[\mathbf{P}]} e_t^{P'} = l_{P'} \quad (5.14)$$

$$\sum_{t \in T[\mathbf{C}]} e_t^{C'} = l_{C'} \quad (5.15)$$

$$e_t^{P'} \leq e_{t+1}^{P'}, \forall t \in T[\mathbf{P}] \quad (5.16)$$

$$e_t^{C'} \leq e_{t+1}^{C'}, \forall t \in T[\mathbf{C}] \quad (5.17)$$

$$0 \leq q_t$$

$$v_t, \bar{v}_t, v_t, v_{C'}, \bar{v}_{P'} \in [\underline{V}, \bar{V}]$$

$$x_t, \bar{a}_t, \underline{a}_t, e_t^{P'}, e_t^{C'} \in \{0, 1\}$$

$$l_{P'} \in \{0, \dots, 64\}, l_{C'} \in \{0, \dots, 32\}$$

## 5.5 Résultats expérimentaux

Cette section est consacrée à une étude computationnelle détaillée de l'application des modèles de PLNE précédant sur une instance réelle typique de petit SDE rural sur un horizon d'une semaine complète. La semaine est découpée en 7 journées types correspondant à jusqu'à 7 problèmes d'optimisation, puis la solution finale est obtenue en reconstruisant le marnage hebdomadaire. Pour chaque solution, la planification de la pompe, le prix de la consommation électrique et le nombre de changements d'état de la pompe sont analysés.

Tous les tests rapportés ont été réalisés en utilisant utilisant Gurobi 9.0.2 et Python 3.6.8 avec sur un processeur quadricœur de 8-threads avec 16 Go de RAM de 2,8 GHz fonctionnant sous Windows 10 (64 bits).

### 5.5.1 Instance de petit SDE rural

Dans nos expériences, nous considérons une configuration réelle de l'installation d'un SDE dans le but de planifier le pompage pour une semaine (e.g. de lundi 6h00 au lundi 5h59 suivant). Nous utilisons des données brutes réelles de demandes de consommation en eau, issues de ses données historiques, sous forme de série temporelle. L'instance considérée comporte un château d'eau d'une capacité de  $\bar{V} = 750m^3$  et est équipée d'une pompe qui peut délivrer  $260m^3$ /heure i.e.  $Q = 65m^3$  sur des pas de temps de 15 minutes. On considère  $\underline{V} = 0m^3$ . Les coefficients de [Équation \(5.1\)](#) sont obtenus à partir de la puissance et du débit effectif de la pompe sur le pas de temps.

Les heures pleines de l'électricité sont en principe de 6h00 à 22h00 puis les heures creuses de 22h00 à 6h00, seuls les dimanches et les jours fériés font exception et sont entièrement en heures creuses. Le coût moyen pour un MWh est d'environ 30€ en heure creuse et environ 42€ en heure pleine, ce coût est soumis à des petites variations de prix durant l'heure pleine et durant l'heure creuse. Finalement, le coût de pompage pour un mètre cube d'eau coûte entre 0.005€ et 0.006€ en heures creuses et entre 0.007€ et 0.008€ en heures pleines.

**Marnage de référence** Le gestionnaire du réseau est très adverse au risque, il a peur des pannes des pompes et de ne pas pouvoir délivrer les utilisateurs, ainsi il met en place deux mécanismes redondants. Premièrement, la station de pompage est équipée de plusieurs pompes mais seulement une seule est active à la fois, les autres sont uniquement des pompes de secours. Deuxièmement, il utilise un marnage de type FTL très serré de très grande valeur pour maintenir un niveau d'eau élevé dans le réservoir à tout moment de la journée. Dans la suite du manuscrit nous appelons ce marnage : "Marnage de référence" (noté REF), qui utilise  $750m^3$  comme marnage haut et  $700m^3$  comme marnage bas durant la journée. Il est évident qu'une telle solution est très coûteuse et de mauvaise qualité.

### 5.5.2 Point de passage et robustesse de RFLATS

Dans le but d'observer et de vérifier la robustesse de RFLATS, on ne s'intéresse ici donc pas directement au coût énergétique ou au nombre de changements d'état de la pompe produits par le marnage, mais seulement à l'évolution du niveau d'eau. Tout d'abord, on choisit une solution (i.e. un marnage de type RFLATS) facilement compréhensible et produisant une évolution du niveau d'eau de bonne qualité, i.e. vidant le réservoir à la fin des heures pleines et remplissant le réservoir à la fin des heures creuses pour presque tous les jours. Puis, on va appliquer ce marnage sur des morceaux de l'instance modifiée. L'idée est de vérifier que si on perd le contrôle du niveau d'eau, alors il sera rapidement corrigé par le marnage de RFLATS. Dans les expérimentations suivantes, la perte de contrôle est modélisée par un changement plus ou moins brutal de la situation initiale. Lorsqu'on observe que le niveau d'eau est assez bien contrôlé, on peut considérer que le niveau passe par des points de passage chaque jour.

La [Figure 5.8](#) montre l'évolution du volume d'eau pour le marnage choisi sur quatre instances un peu modifiées. L'évolution du niveau d'eau pour les 4 instances est étudiées sur trois parties de 40 heures chacune, les autres périodes de temps de la semaine sont visuellement blanchies. Chaque partie est composée d'un **P** puis d'un **C** puis d'un **P**. Le marnage haut de **P'** choisi durant les trois parties est aussi différent, on a pour :

- la première partie, un **P'** avec un marnage serré ( $200m^3$ ), puis un deuxième **P'** avec un marnage lâche ( $600m^3$ ),

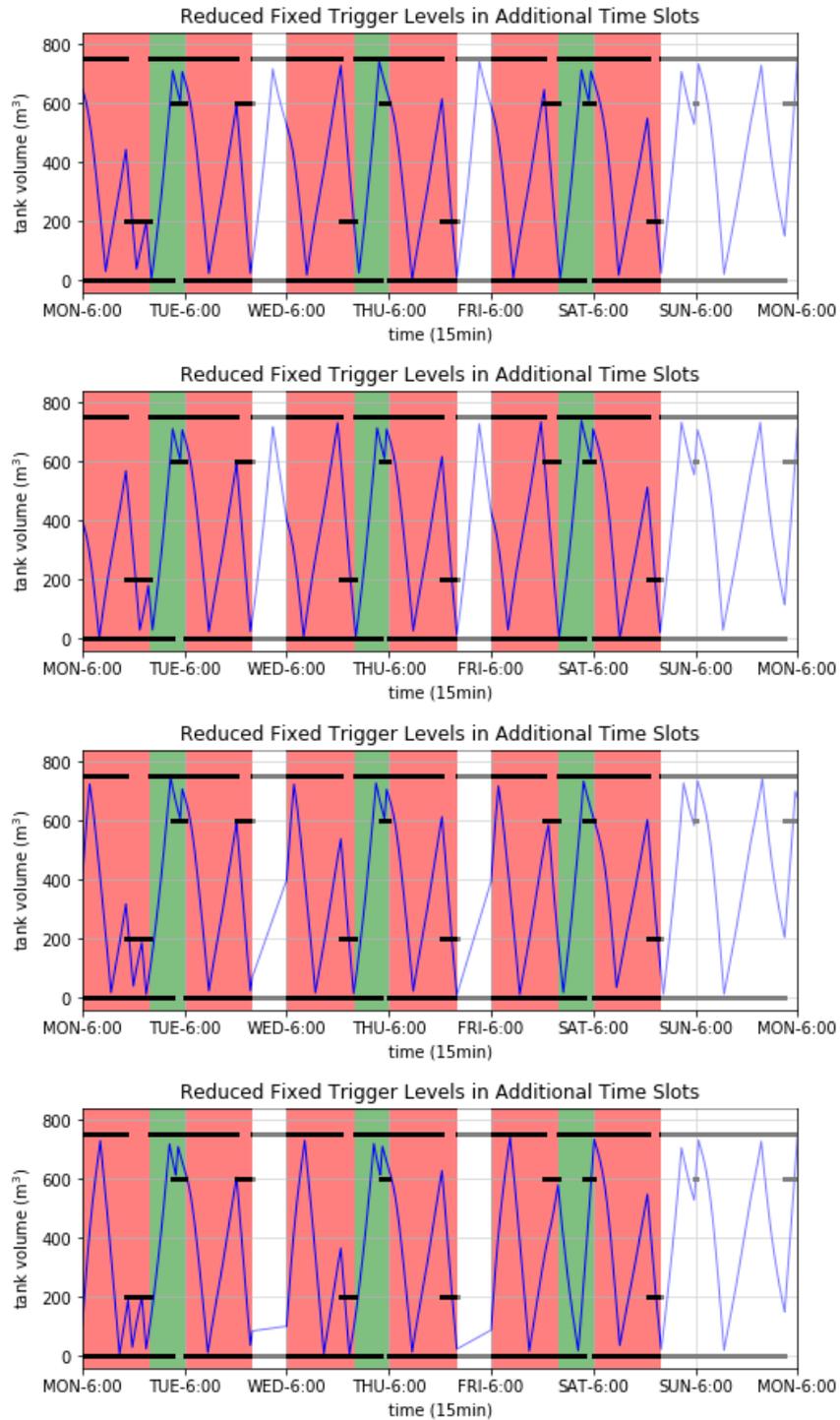


FIGURE 5.8 – Solution de RFLATS sur des instances modifiées

- la deuxième partie, un  $\mathbf{P}'$  avec un marnage serré ( $200m^3$ ), puis un deuxième  $\mathbf{P}'$  avec un marnage serré ( $200m^3$ ),
- la dernière partie, un  $\mathbf{P}'$  avec un marnage lâche ( $600m^3$ ), puis un deuxième  $\mathbf{P}'$  avec un marnage serré ( $200m^3$ ).

Pour rappel, un marnage serré (resp. lâche) est un marnage où la différence entre le marnage haut et le marnage bas est faible (resp. forte), et indépendamment de la durée de la période où elle s'applique.

La première sous-figure montre la solution sur l'instance d'une semaine puis les trois autres sous-figures montrent ce marnage sur des instances un peu modifiées afin que la situation initiale de chaque partie correspondent à, dans l'ordre :

1. un niveau d'eau initial moyen ( $400m^3$ ) et que la pompe soit éteinte,
2. un niveau d'eau initial moyen ( $400m^3$ ) et que la pompe soit allumée,
3. un niveau d'eau initial faible ( $100m^3$ ) et que la pompe soit allumée,

Les courbes présentées dans la [Figure 5.8](#) suggèrent les commentaires suivants :

- Lundi/mardi : Grâce au marnage serré, peu importe la situation initiale, le niveau d'eau à la fin de  $\mathbf{P}$  est déjà contrôlé (entre 0 et  $100m^3$ ). Les comportements durant  $\mathbf{C}$  sont assez similaire. Puis, le marnage lâche produit peu d'écart, à chaque fois le mardi à 22h00, on retrouve bien le réservoir plutôt vidé (entre 20 et  $80m^3$ ).
- Mercredi/jeudi : Grâce au marnage serré, peu importe la situation initiale, le niveau d'eau à la fin de  $\mathbf{P}$  est déjà contrôlé (entre 0 et  $100m^3$ ). Puis, le marnage serré d'après contrôle encore le niveau, à chaque fois le jeudi à 22h00, on retrouve bien le réservoir vidé (entre 0 et  $20m^3$ ).
- Vendredi/samedi : Le marnage lâche autorise le niveau d'eau à la fin de  $\mathbf{P}$  à prendre des valeurs dans une grande plage (e.g. entre 200 et  $600m^3$ ). Le niveau est un peu corrigé par le marnage serré de  $\mathbf{C}$ , puis totalement corrigé le samedi à 22H00 par son marnage serré.

De manière générale, même si la situation initiale est largement différente, on remarque qu'un marnage serré de type RFLATS permet d'emmener le niveau d'eau au point de passage, ainsi la période d'après commence avec un niveau plutôt correct par rapport à ce qui était prévu. Dans des cas extrêmes, le niveau est totalement corrigé seulement après deux marnages serrés. Dans le contexte de la minimisation du coût énergétique et du nombre de changements d'état de

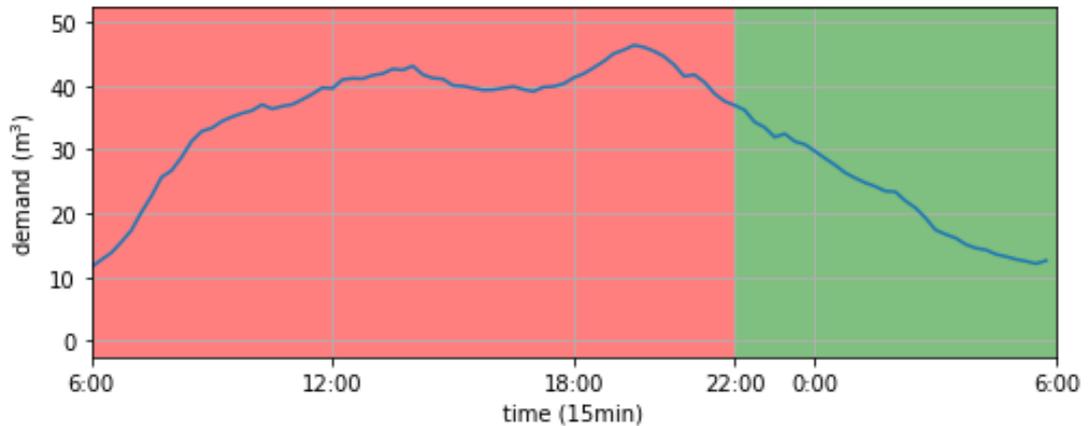


FIGURE 5.9 – Demande pour la journée type : jour de semaine

la pompe par automatisation du pompage, uniquement la gestion avec RFLATS peut produire des solutions ayant un marnage serré sur une courte période. Ainsi, lorsqu'on garantit l'utilisation d'un marnage serré de type RFLATS, on peut se permettre de découper l'horizon au niveau des points de passage, par exemple tous les jours à 6h00 le niveau est au maximum du réservoir.

Des journées types sont déterminées via l'historique des données en appliquant des moyennes, et nommées par : jour de semaine, samedi, dimanche, jour férié... La planification recherchée sera finalement déterminée par la solution des sous-problèmes sur les journées types prises séparément puis reconstruite après les optimisations.

### 5.5.3 Journée type : jour de semaine

Les jours de semaine sont composés de deux parties, les **P** de 6h00 à 22h00 puis les **C** de 22h00 à 6h00. La Figure 5.9 montre la demande considérée sur des pas de temps de 15 minutes pour la journée type : jour de semaine. Principalement, on y observe un comportement de jour (de 10h00 à 22h00), un comportement de nuit (de 3h00 à 7h00) et un comportement de levé/couché (de 7h00 à 10h00 et 22h à 3h00). Durant la journée, il y a des demandes d'environ  $40m^3$  durant les pas de temps de 15 minutes, avec de petits pics à 14h00 ( $44m^3$ ) et un plus grand pic à 19h00 ( $47m^3$ ). Durant la nuit, les demandes sont très faibles, de 13 à  $16m^3$ . La consommation journalière totale est d'environ  $3000m^3$ , ainsi la pompe doit être active durant environ la moitié du temps (car  $3000 / 250 = 12$  heures) afin de répondre à la demande.

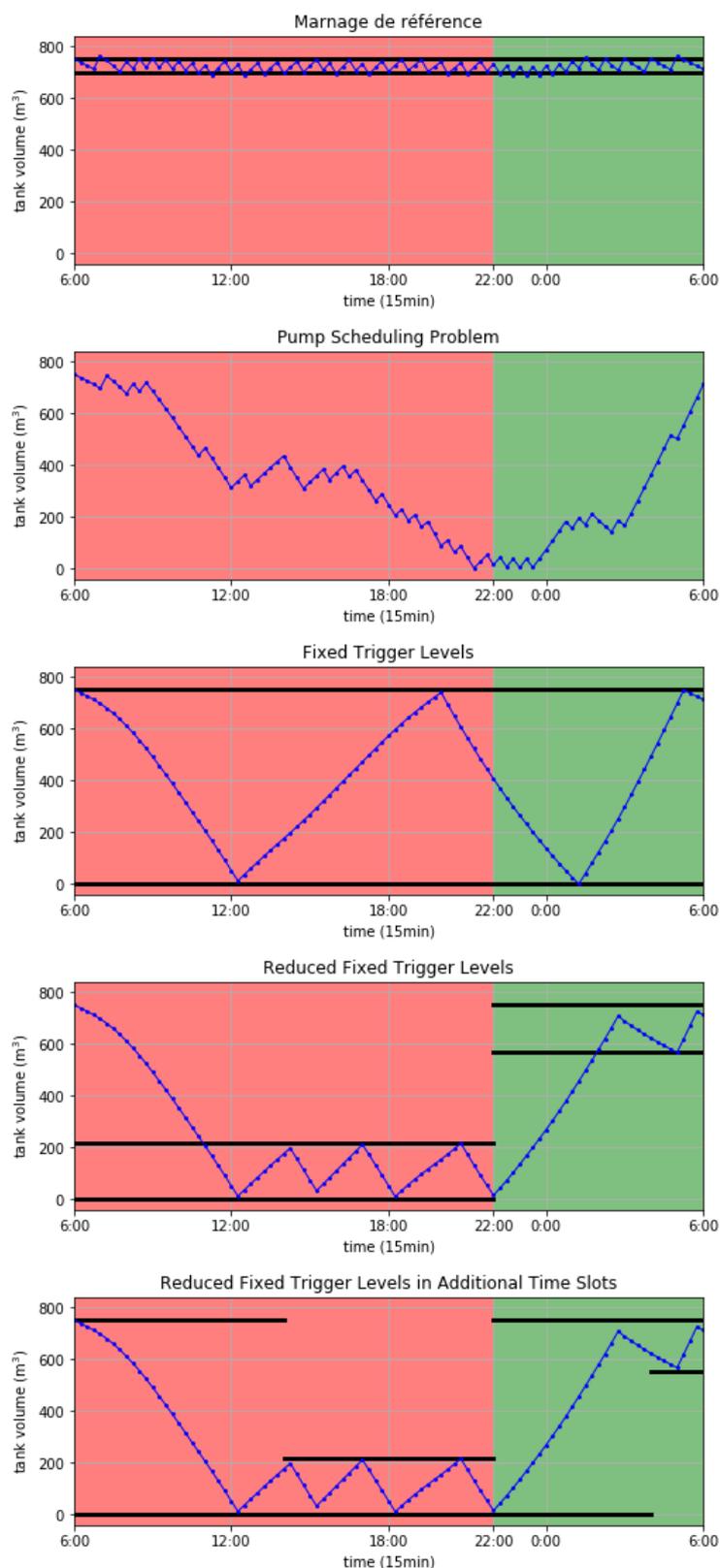


FIGURE 5.10 – Solution optimale pour la journée type : jour de semaine

PLNE	€	NCE	CPU	$\underline{v}$	$\bar{v}$
REF	21.1296	70	-	700	750
PSP	18.7562	50	16ms	-	-
FTL	20.2527	5	-	-	-
RFTL	19.3445	10	178ms	567	216
RFLATS	19.3445	10	262ms	554	216

TABLEAU 5.1 – Résultats pour la solution de référence REF et les solutions optimales obtenues avec PSP, FTL, RFTL et RFLATS sur la journée type : jours de semaine

La [Figure 5.10](#) montre l'évolution du volume d'eau dans le château d'eau dans le temps pour les solutions optimales obtenues avec REF, FTL, RFTL et RFLATS sur la journée type : jours de semaine. La courbe et les points bleus représentent le volume d'eau dans le château d'eau (en  $m^3$ ). Les lignes noires représentent les marnages (en  $m^3$ ) leurs valeurs sont montrées dans le [Tableau 5.1](#). La zone en rouge représente les **P**, tandis que la zone en vert représente les **C**.

Le [Tableau 5.1](#) affiche les résultats pour la solution de référence REF et les solutions optimales obtenues avec PSP, FTL, RFTL et RFLATS sur la journée type : jours de semaine. Pour chaque problème, la colonne intitulée '€' indique le coût énergétique en euros de [Équation \(5.2\)](#) ; la colonne intitulée 'NCE' indique le nombre de changements d'état de la pompe de [Équation \(5.2\)](#) ; la colonne intitulée 'CPU' indique le temps cpu total résultant ; les colonnes intitulées  $\underline{v}$  et  $\bar{v}$  indiquent les valeurs des marnages bas et des marnages hauts qu'on peut observer dans la [Figure 5.10](#) qui ne sont pas les constantes  $\underline{V}$  et  $\bar{V}$ .

Les courbes présentées dans la [Figure 5.10](#) et les résultats présentés dans le [Tableau 5.1](#) suggèrent les commentaires suivants :

- La valeur 21.1296€ de REF est un très mauvais exemple de gestion, elle produit le grand nombre de changement d'état de la pompe (NCE = 70). Son avantage repose uniquement sur le fait qu'elle permet de garantir un niveau d'eau très élevé.
- La valeur 18.7562€ de PSP est le meilleur coût qu'on puisse espérer pour la gestion de l'instance. Pour rappel, cette gestion nécessite un contrôleur avancé. La solution est obtenue en échange d'un certain nombre de changements d'état de la pompe (NCE = 50), afin de ne pas pomper durant les quelques heures coûtant les plus cher durant **P** et **C**. Notons que dans les formulations usuelles de PSP, le nombre de changements d'état de la pompe

est souvent contrôlé et borné.

- La valeur 20.2527€ de FTL correspond à une solution de très mauvaise qualité. La solution ne propose pas gain économique, on remarque qu'une grande quantité d'eau est pompée durant  $\mathbf{P}$ .
- Le réservoir vidé à 22h00 correspond à un temps de pompage faible durant  $\mathbf{P}$ . Les courbes de PSP, RFTL et RFLATS ont le volume d'eau proche de  $0m^3$  à 22h00 (i.e.  $16.1m^3$ ). Notons que l'écart entre  $0m^3$  et  $16.1m^3$  est présent car, dans la formalisation du problème, la pompe ne peut délivrer que  $0m^3$  ou  $65m^3$  sur chaque pas de temps de 15 minutes.
- Le marnage bas de RFLATS de  $554m^3$  de 4h45 à 6h00 permet d'obtenir le réservoir "rempli" à 6h00. Notons que deux autres combinaisons donnent strictement le même résultat :
  1. en appliquant n'importe quelle valeur entre  $554m^3$  et  $750m^3$  de 4h45 à 6h00,
  2. en appliquant la valeur  $554m^3$  sur une plage commençant entre 22h00 et 4h45 puis se terminant à 6h00.
- Bien que les marnages sont légèrement différents, la solution de RFTL et de RFLATS donnent les mêmes variations. Ainsi, on ne voit pas d'intérêt direct de RFLATS par rapport à RFTL dans cette instance. Mais on observe bien que RFLATS offre plus de liberté que RFTL, car RFTL est un cas spéciale de RFLATS.

On peut ajouter un commentaire global à l'utilisation des marnages ; en regardant les figures on comprend qu'il n'est pas facile de prévoir si changer un peu la valeur d'un des marnage va augmenter ou diminuer la valeur de l'objectif, le voisinage des optimums locaux est très chaotique. Finalement, on se rend compte qu'il est compliqué de traduire la fonction objectif en faisant apparaître uniquement les variables de décision  $l_{\mathbf{P}'}$ ,  $l_{\mathbf{C}'}$ ,  $\bar{v}_{\mathbf{P}'}$  et  $\underline{v}_{\mathbf{C}'}$ .

Les expérimentations montrent bien que RFLATS est plus intéressant que FTL, et plus rapide que RFTL. La solution est aussi facilement applicable sur les automates en définissant les plages horaires appropriées. Comparer à la méthode par programmation dynamique qu'on propose dans [Sous-section 5.3.3](#), il est plus facile pour un gestionnaire du réseau de comprendre les variables d'un solveur de PLNE que de regarder un graphe où les sommets représentent un ensemble de solution. L'approche PLNE a aussi l'avantage d'intégrer de nouvelles contraintes plus facilement (e.g. celles pour le multi-réservoir ou multi-pompe).

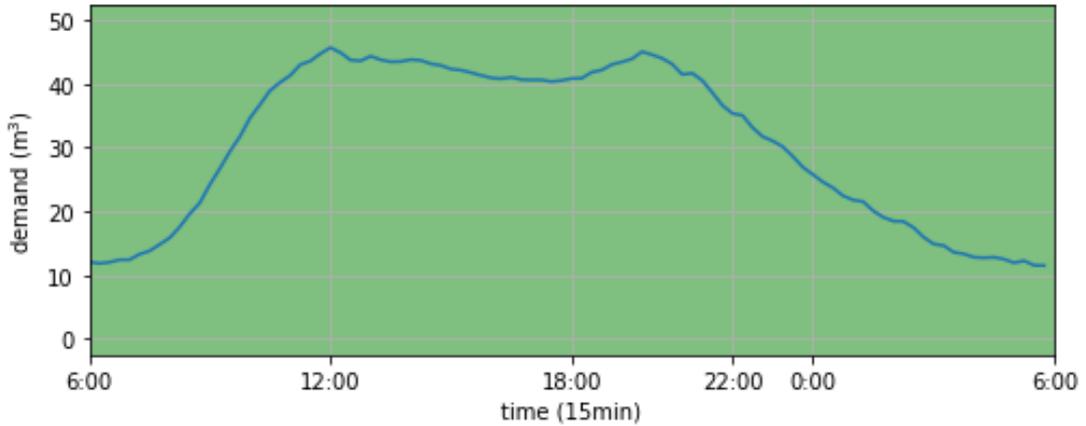


FIGURE 5.11 – Demande pour la journée type : dimanche

PLNE	€	NCE	CPU
REF	15.3490	62	-
PSP	14.6606	47	1.4ms
FTL	16.1689	4	-
RFTL	15.2355	7	72ms
RFLATS	15.2232	8	67ms

TABLEAU 5.2 – Résultats pour la solution de référence REF et pour les solutions optimales obtenues avec PSP, FTL, RFTL et RFLATS sur la journée type : dimanche

#### 5.5.4 Journée type : dimanche

Les dimanches sont composés seulement de **C**, donc de 6h00 à 6h00 (du lendemain). La [Figure 5.11](#) montre la demande considérée sur des pas de temps de 15 minutes pour la journée type : dimanche. Principalement, on y observe un comportement de jour (de 12h00 à 22h00), un comportement de nuit (de 3h00 à 8h00) et un comportement de levé/couché (de 8h00 à 10h00 et 22h à 3h00). Donc dans la globalité le comportement est similaire à celui des jours de semaine. La principale différence est que l'augmentation progressive de la demande de 7h00 à 10h00 est décalée entre 10h00 et 13h00. La consommation journalière totale est d'environ  $3000m^3$ , comme pour les jours de semaine.

Le [Tableau 5.2](#) affiche les résultats pour la solution de référence REF et pour les solutions optimales obtenues avec PSP, FTL, RFTL et RFLATS sur la journée type : dimanche. Les résultats présentés dans le [Tableau 5.2](#) suggèrent des commentaires similaires avec ceux pour la journée type jours de semaine. Notons

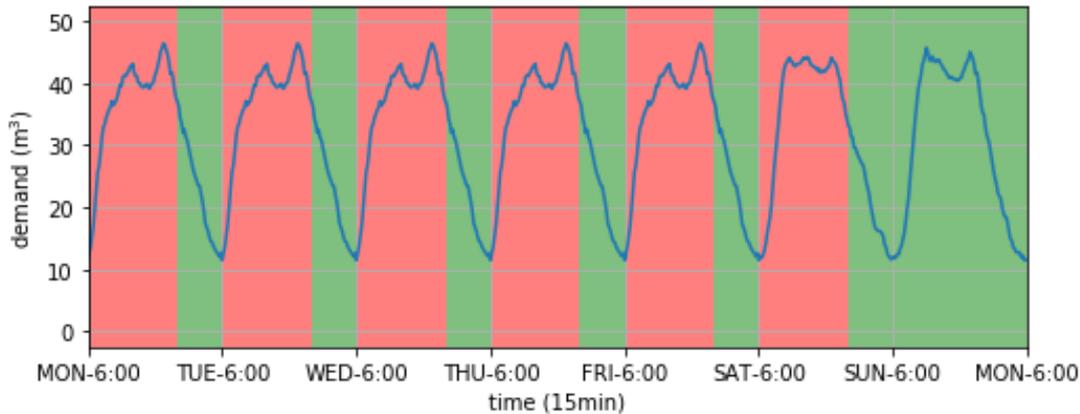


FIGURE 5.12 – Demande pour la semaine

simplement que RFLATS est meilleure que RFTL sur cette instance.

On remarquera par la suite que les expérimentations pour les autres journées types produiront aussi des commentaires similaires.

### 5.5.5 Marnage reconstruit

Dans cette sous-section, la semaine qu'on considère est composée de cinq fois le même jour de semaine puis un samedi puis un dimanche, la semaine commençant ainsi un 'Lundi' à 6h00 et se termine le 'Lundi' à 5h59 de la semaine d'après.

La Figure 5.12 montre la demande considérée pour la semaine sur des pas de temps de 15 minutes. On peut y retrouver les demandes des jours de semaine de la Figure 5.9 et du dimanche de la Figure 5.11. Les indices de temps des modèles de la section précédente sont un peu adaptés pour obtenir l'horizon de la semaine à partir de 7 instances d'une journée.

La Figure 5.13 montre l'évolution du volume d'eau pour les solutions obtenues avec REF, RFLATS et RFLATSre. La courbe et les points bleus représentent le volume d'eau dans le château d'eau (en  $m^3$ ). Les lignes noires représentent les marnages (en  $m^3$ ) leurs valeurs sont montrées dans le Tableau 5.3. La zone en rouge représente les **P**, tandis que la zone en vert représente les **C**. La situation initiale est identique pour les trois courbes, i.e. le réservoir est rempli le lundi à 6h00.

Le Tableau 5.3 affiche les résultats pour la solution de référence REF et pour les solutions obtenues avec RFLATS et RFLATSre. RFLATSre correspond à la solution RFLATS reconstruit à partir des solutions optimales obtenues sur 7

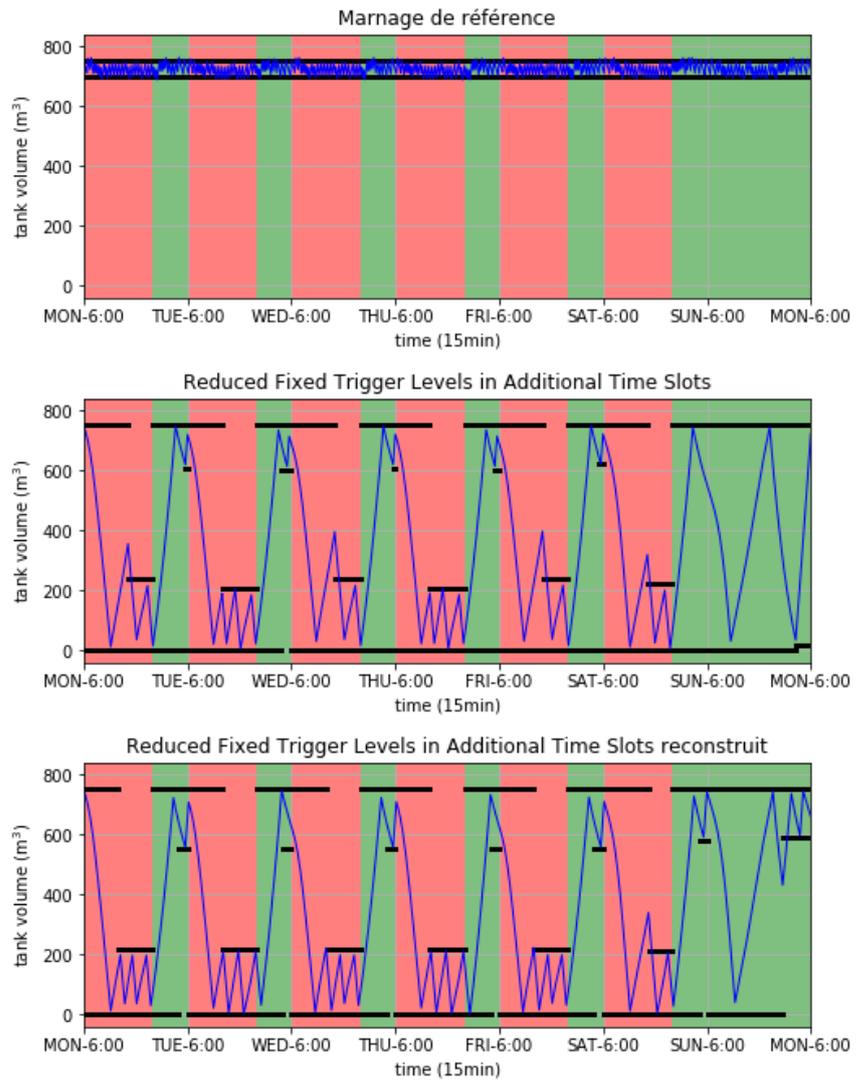


FIGURE 5.13 – Solutions optimales pour la semaine

PLNE	€	NCE	CPU
REF	146.3357	458	-
RFLATS	136.2712	53	95sec
RFLATSre	137.4885	60	798ms

TABLEAU 5.3 – Résultats pour la solution de référence REF et pour les solutions obtenues avec RFLATS et RFLATSRe sur la semaine

problèmes résolus séparément. Pour chaque problème, la colonne intitulée '€' indique le coût énergétique en euros de [Équation \(5.2\)](#); la colonne intitulée 'NCE' indique le nombre de changements d'état de la pompe de [Équation \(5.2\)](#); la colonne intitulée 'CPU' indique le temps cpu total de l'exécution.

Les courbes présentées dans la [Figure 5.13](#) et les résultats présentés dans le [Tableau 5.3](#) suggèrent les commentaires suivants :

- Pour REF, la valeur 146.3357€ est un mauvais coût pour la gestion de l'instance. De plus elle génère un très grand nombre de changements d'état de la pompe (NCE = 458).
- Pour RFLATS, le problème est résolu en 95 secondes. Sa solution de faible coût est un marnage plutôt serré. Le volume d'eau est proche de  $0m^3$  tous les jours à 22h00 et proche de  $\bar{V}$  tous les jours à 6h00.
- Pour RFLATSre, sa solution correspond aux différents marnages obtenus pour la résolution des journées types, et est obtenue en seulement 798ms. La courbe pour RFLATSre a son volume d'eau plutôt proche de  $0m^3$  presque tous les jours à 22h00.
- Pour RFLATS, on remarque que les courbes de lundi, mercredi et vendredi ont des comportements similaires (e.g. le pic au début de l'heure additionnelle), puis on remarque que celles de mardi et jeudi ont aussi des comportements similaires (e.g. 3 cycles de remplissage). Pour RFLATSre, on retrouve aussi des comportements similaires entre ces mêmes jours : pour lundi, mercredi et vendredi, les courbes durant **P** et **C** se ressemblent ; pour mardi et jeudi, les courbes ressemblent à celle de RFLATS. Le samedi est globalement le même entre RFLATS et RFLATSre. Les écarts entre RFLATS et RFLATSre s'explique par le fait qu'on ne peut pas atteindre parfaitement le  $0m^3$  ou le  $\bar{V}m^3$ , i.e. la situation initiale qu'on prévoit chaque jour est légèrement différente. Cependant, le dimanche est bien différent entre RFLATS et RFLATSre, on peut en partie expliquer cela par l'accumulation

d'écart durant les 6 premiers jours.

Finalement, RFLATS et RFLATSre retournent des solutions très rapidement et les solutions sont assez similaires, mais RFLATSre est largement plus rapide (environ 1sec contre 100 sec). Nous accordons un intérêt majeur à la reconstruction du marnage car elle permet de travailler sur des problèmes de plus petites tailles et donc plus simples, sans trop perdre de précision. Dans la suite du manuscrit, on va proposer une version robuste du problème où le temps de calcul explose, ce qui ne respecte pas notre limite de 8 heures maximales pour la résolution du problème.

## 5.6 Résumé du chapitre

Nous présentons le problème de la minimisation du coût énergétique généré par les opérations de pompage pour alimenter un réservoir d'eau surélevé à partir d'une source dans les systèmes de distribution d'eau. Le problème d'optimisation des opérations de pompage, connu sous le nom générique de *Pump Scheduling Problem*, est appliqué grâce à un contrôleur avancé. Le marnage est une stratégie 'automatisée', où les pompes se mettent automatiquement en marche ou à l'arrêt lorsque le niveau d'eau atteint une valeur donnée dans le château d'eau.

Nous avons re-décrit les modèles linéaire en nombres entiers pour les problèmes de réseau d'eau : le *Pump Scheduling Problem*, le marnage de type *Fixed trigger Levels* et le marnage de type *Reduced Fixed trigger Levels*. Nous montrons la complexité polynomiale avec la programmation dynamique et proposons un premier modèle linéaire en nombres entiers pour le problème avec l'utilisation des marnages de type *Reduced Fixed trigger Levels in Additional Time Slots* (RFLATS), initialement proposé par [88], en utilisant les contraintes de nos propres modélisations des trois autres problèmes. La fonction objectif du problème prend en compte la minimisation de la consommation d'électricité pour l'activation des pompes et une certaine pénalité si le réservoir n'est pas rempli en fin d'étude. Les résultats numériques montrent que le RFLATS apporte un meilleur coût énergétique comparé à un marnage de référence, qui est un marnage garantissant une forte autonomie en cas de panne.

Dans [88], le marnage de type RFLATS obtient sont importance pour les économies qu'il apporte. Dans nos travaux, ce marnage nous intéresse particulièrement pour son utilisation d'un marnage serré durant les périodes additionnelles,

mais Ce type de marnage donne une certaine garantie en guidant le niveau d'eau vers des points de passages à la fin des heures pleines et heures creuses. C'est pourquoi, on peut plus ou moins garantir que tous les jours à 6h00 le réservoir sera rempli, ainsi la robustesse de RFLATS permet de découper l'horizon en petits problèmes correspondant chacun à une journée type, ainsi la solution finale est obtenue en recomposant le résultat des optimisations sur les journées types prises séparément.

# Chapitre 6

## Modèle robuste pour RFLATS avec des incertitudes sur les demandes

### Sommaire

---

6.1	Formulation robuste pour RFLATS . . . . .	<b>102</b>
6.1.1	Ensemble d'incertitude . . . . .	102
6.1.2	Formulation du problème robuste . . . . .	103
6.2	Résolution de la version robuste de RFLATS . . . . .	<b>105</b>
6.2.1	Détermination des marnages horaires . . . . .	105
6.2.2	Évaluation d'un marnage horaire donné . . . . .	106
6.2.3	Graphe de plus long chemin pour la recherche de la solution pire cas . . . . .	107
6.2.4	Réduction du graphe de la recherche du pire cas . . . . .	108
6.2.5	Simplification du graphe de la recherche du pire cas . . . . .	110
6.2.6	Algorithme pour la solution robuste . . . . .	112
6.3	Résultats expérimentaux . . . . .	<b>113</b>
6.3.1	Instance de petit SDE rural . . . . .	113
6.3.2	Résultats numériques pour la recherche du pire cas . . . . .	114
6.3.3	Résultats expérimentaux par résolution du problème robuste . . . . .	119
6.3.4	Résultats expérimentaux pour le marnage reconstruit et prix de la robustesse . . . . .	122
6.4	Résumé du chapitre . . . . .	<b>124</b>

---

Nous abordons dans ce chapitre une version robuste du problème de minimisation du coût énergétique généré par les opérations de pompage de type RFLATS du Chapitre 5. Pour chaque instance, on définit des incertitudes sur la demande

(la consommation). On utilise la méthodologie *Seasonal and Trend decomposition using Loess* de Cleveland, R. B., Cleveland, W. S., McRae, J. E., et Terpenning, I. (1990) [31] et un concept d'ensemble d'incertitude similaire à celui de Bertsimas, D., et Sim, M. (2004) [15]. Le problème robuste est résolu par énumération des marnages possibles. Pour chaque marnage possible, on détermine le pire cas avec un algorithme de plus long chemin dans un graphe séquentiel sans circuit représentant l'évolution du niveau d'eau. Afin de respecter la durée d'exécution maximale autorisée, on propose aussi une méthode approchée trouvant souvent le pire cas grâce à une simplification du graphe

## 6.1 Formulation robuste pour RFLATS

### 6.1.1 Ensemble d'incertitude

La demande est considérée comme une série temporelle dans laquelle chaque échantillon est collecté toutes les 15 minutes (96 mesures par jour). Similairement à l'ensemble d'incertitude du Chapitre 4, on applique la décomposition STL [31] pour décomposer la série temporelle afin de récupérer la dite "donnée traitée" et les informations du résiduel.

La donnée traitée  $c_t$  est obtenue en sommant la composante de la tendance et la composante de la saisonnalité, i.e.  $c_t = trend(t) + seasonal(t)$ . On peut aussi travailler avec une prédiction de  $trend(t)$  et/ou de  $seasonal(t)$ , afin de récupérer des consommations prédites. La composante résiduelle *remainder* peut être considérée comme les réalisations d'une variable aléatoire, où la plupart des données brutes tombent entre  $c_t - 3\sigma$  et  $c_t + 3\sigma$ , tel que  $\sigma$  soit l'écart type de *remainder*.

Pour chaque journée type, on note  $d = (d_1, \dots, d_{|T|})$  le vecteur des variables de déviations possibles du problème robuste, où les déviations sont typiquement bornées comme  $-b \leq d_t \leq b$  tel que  $b = r * \sigma$  avec  $r = 3$ . Les déviations peuvent changer le coût de trois manières (impacte de coût du plus fort au plus faible) : changer la quantité de pompage, charger la pénalité de réservoir non rempli et décaler les moments de pompage. Les déviations positives peuvent augmenter directement la quantité de pompage alors que les déviations négatives réduisent largement cette quantité. Ces deux types de déviations peuvent changer la pénalité de réservoir non rempli, cette pénalité est bornée en fonction de la taille du réservoir.

Ainsi, dans un contexte d'optimisation robuste, nous allons contraindre les écarts à être bornés comme  $0 \leq d_t \leq b$ . Le paramètre  $B$  est le budget d'incertitude tel que  $\sum_{t \in T} d_t \leq B$ . Il doit généralement être du même ordre que  $\sqrt{T}$ , ainsi on choisit la formulation  $B = R * r\sigma$  avec  $R \approx 3 * \sqrt{T}$ .

### 6.1.2 Formulation du problème robuste

La version robuste du problème se base sur la formulation déterministe du marnage de type RFLATS : (5.2-5.8, 5.9''-5.12'', 5.13-5.17) du Chapitre 5. On remplace la fonction de coût Équation (5.2), en incluant la maximisation sur  $d$ , avec :

$$\min_{l_{\mathbf{P}'}, l_{\mathbf{C}'}, \bar{v}_{\mathbf{P}'}, \underline{v}_{\mathbf{C}'}} \max_d \sum_{t \in T} C_t * PC(q_t) + C_{\mathbf{P}} * PC(\bar{V} - v_T) \quad (6.1)$$

**Non-convexité de la fonction de coût** Il est compliqué de traduire la fonction de coût en faisant apparaître uniquement les variables de décision ( $l_{\mathbf{P}'}, l_{\mathbf{C}'}, \bar{v}_{\mathbf{P}'}, \underline{v}_{\mathbf{C}'}$  et  $d$ ). Dans la Figure 6.1, montre des tentatives d'interpolation de la fonction de coût de la version déterministe entre deux des points. Nous faisons évoluer un paramètre  $\lambda$ , avec un pas de 0.01, afin d'obtenir des points en utilisant une combinaison convexe des deux points, puis en évaluant le coût de ce point. Pour chaque courbe, on choisit 2 points parmi 4 points : un marnage très serré (130, 620, 32, 16), un marnage serré (190, 550, 32, 16), un marnage neutre (351, 324, 27, 1) et un marnage lâche (700, 50, 4, 4).

Les courbes révèlent par exemple que la fonction objective n'est pas convexe ou que le minimum semble plutôt être parmi les solutions représentant des marnages serrés, mais elles révèlent principalement que le voisinage des optimums locaux est très chaotique. Il est donc compliqué de savoir à l'avance si une solution est un optimum ou optimum local sans évaluer toutes les solutions du voisinage. Finalement, on peut faire les même constatations pour la fonction de coût de la version robuste du problème.

On remplace aussi la contrainte Équation (5.3), la conservation de flot, par un équivalent prenant en compte  $d$  par :

$$v_{t+1} = v_t + q_t - D_t - d_t, \forall t. \quad (6.2)$$

Finalement, pour définir l'ensemble d'incertitude, il faut ajouter les contraintes

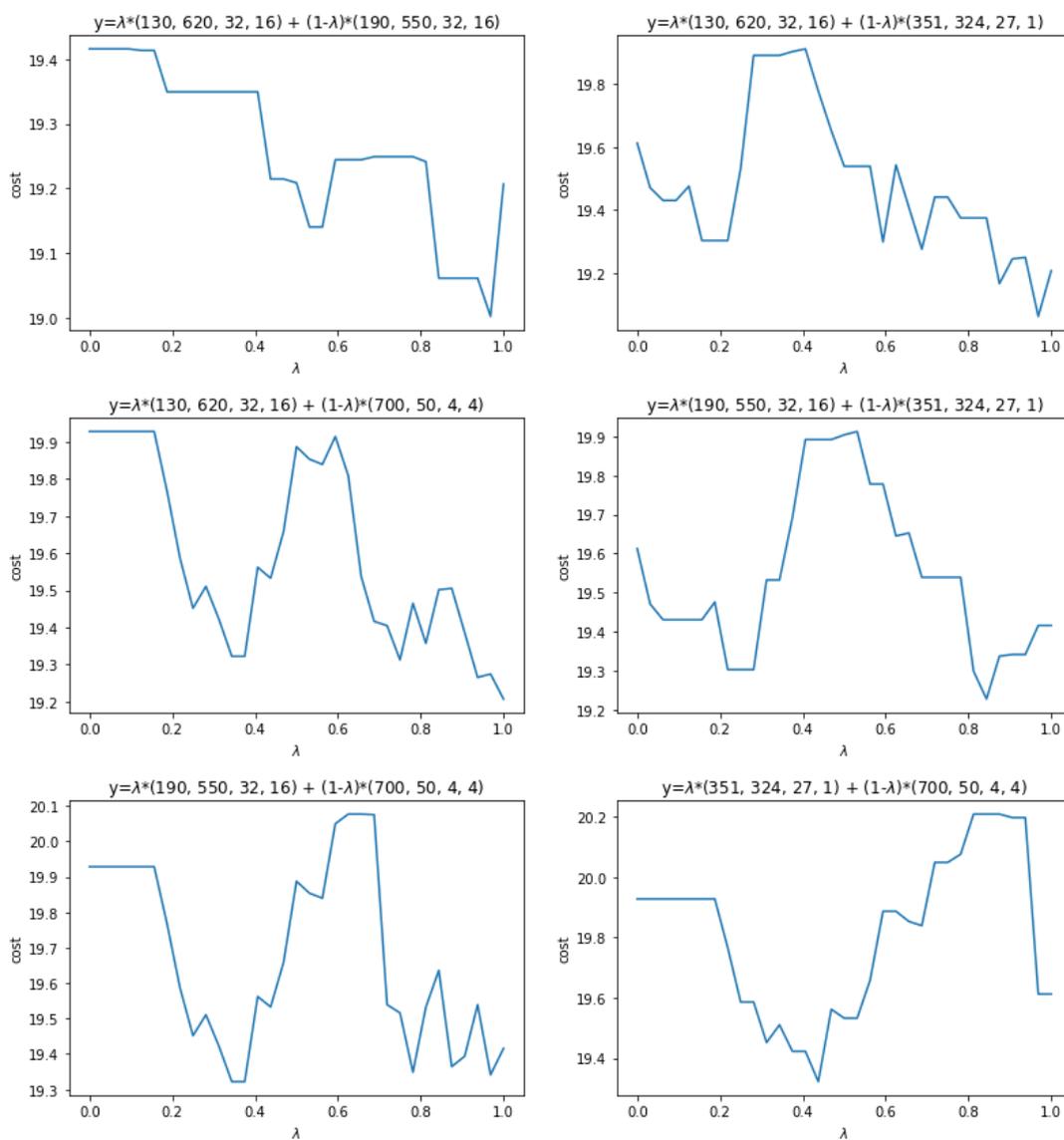


FIGURE 6.1 – Non-convexité de la fonction de coût du problème déterministe

suivantes :

$$0 \leq d_t \leq r\sigma, \forall t \quad (6.3)$$

$$\sum_{t \in T} d_t \leq B. \quad (6.4)$$

Cette formulation fait donc intervenir un *minmax* ce qui rend la résolution de la version robuste plus compliquée que celle de la version déterministe. Ce problème fait penser à la programmation bi-niveaux avec deux niveaux de décision, avec un programme Leader de minimisation (avec  $l_{\mathbf{P}'}, l_{\mathbf{C}'}, \bar{v}_{\mathbf{P}'}, \underline{v}_{\mathbf{C}'}$ ) et un programme Suiveur de maximisation (avec  $d$ ), qui sont souvent difficiles à résoudre.

Dans ce qui suit, le temps d'exécution est limité à 8 heures par instance, ainsi nous n'allons pas utiliser le programme mathématique. À la place, on propose une combinaison utilisant une méthode d'énumération pour la minimisation liée au choix des marnages (pour  $l_{\mathbf{P}'}, l_{\mathbf{C}'}, \bar{v}_{\mathbf{P}'}, \underline{v}_{\mathbf{C}'}$ ), et combinée avec une méthode de programmation dynamique permettant la résolution du pire cas pour un marnage donné (pour  $d$ ).

## 6.2 Résolution de la version robuste de RFLATS

Dans cette section, nous traiterons la résolution de la version robuste de RFLATS pour la journée type "jour de semaine". Le même type de réflexion et d'algorithmes pourront être développés pour les autres journées types ("samedi", "dimanche", "jour férié" ...).

### 6.2.1 Détermination des marnages horaires

On appelle "marnage horaire", la traduction des variables de décision des différents problèmes (FTL, RFTL, RFLATS) en  $T$  variables plus facilement manipulable dans nos algorithmes (i.e. une variable par pas de temps). L'Algorithme 6.1 est un exemple d'implémentation pour RFLATS en se basant sur ces quatre variables de décision d'un RFLATS déterministe (cf. paragraphe 5.4.3) :  $l_{\mathbf{P}'}, l_{\mathbf{C}'}, \bar{v}_{\mathbf{P}'}$  et  $\underline{v}_{\mathbf{C}'}$ .

---

**Algorithme 6.1** Déterminer les marnages d'un RFLATS

---

```

1: fonction  $\bar{v}, \underline{v} \leftarrow \text{RETRIEVERFLATS}(l_{\mathbf{P}'}, l_{\mathbf{C}'}, \bar{v}_{\mathbf{P}'}, \underline{v}_{\mathbf{C}'})$ 
2:    $\bar{v} \leftarrow [\bar{V}, \dots, \bar{V}]$ 
3:   pour  $t$  de  $T_{\mathbf{P}} - l_{\mathbf{P}'}$  à  $T_{\mathbf{P}}$  faire
4:      $\bar{v} \leftarrow \bar{v}_{\mathbf{P}'}$ 
5:   fin pour
6:    $\underline{v} \leftarrow [\underline{V}, \dots, \underline{V}]$ 
7:   pour  $t$  de  $T_{\mathbf{C}} - l_{\mathbf{C}'}$  à  $T_{\mathbf{C}}$  faire
8:      $\underline{v} \leftarrow \underline{v}_{\mathbf{C}'}$ 
9:   fin pour
10:  retourner  $\bar{v}, \underline{v}$ 
11: fin fonction

```

---



---

**Algorithme 6.2** Pseudo code pour l'évaluation d'un marnage

---

```

1: fonction  $cost, v \leftarrow \text{EVAL}(\bar{v}, \underline{v}, x_{init}, v_{init}, t1, t2, d = (0, \dots, 0))$ 
2:    $cost, x, v_{t1} \leftarrow 0, x_{init}, v_{init}$ 
3:   pour  $t$  de  $t1$  à  $t2$  faire
4:      $v' \leftarrow v + x * Q - D_t - d_t$ 
5:     si  $v' < \underline{v}$  ou  $\bar{v} < v'$  alors
6:        $x \leftarrow \neg x$ 
7:     fin si
8:      $v_{t+1} \leftarrow v_t + x * Q - D_t - d_t$ 
9:      $cost \leftarrow cost + x * C_t * PC(Q)$ 
10:  fin pour
11:  si  $t2 == T$  alors
12:     $cost \leftarrow cost + C_{\mathbf{P}} * PC(\bar{V} - v_T)$ 
13:  fin si
14:  retourner  $cost, v$ 
15: fin fonction

```

---

### 6.2.2 Évaluation d'un marnage horaire donné

On appelle "niveau d'eau horaire", le vecteur représentant l'évolution du niveau d'eau d'une solution. On peut coder une fonction permettant de déterminer les niveaux d'eau horaires d'un marnage horaire, en calculant les activations de la pompe de  $t = 1$  à  $T = T$  pour le marnage donné. On peut ainsi aussi évaluer le marnage et obtenir son coût. Par exemple, l'Algorithme 6.2 retourne le coût du pompage et le vecteur des niveaux d'eau horaires associés. Pour être plus générique, cette algorithme prend aussi en entrée l'état initial de la pompe  $x_{init}$ , l'état initial le niveau d'eau  $v_{init}$ , et la période d'étude est renseignée par les indices  $t1$  et  $t2$ . On permet aussi de fournir le vecteur des déviations  $d$  si nécessaire.

La ligne 2 de l'algorithme initialise le coût total  $cost$ , l'état de la pompe  $x$  et du niveau d'eau  $v$  au début de la période d'étude. Ensuite à chaque itération  $t$ , on calcul  $v'$ , le niveau d'eau du réservoir à la fin de  $t$  si on ne change pas l'état de la pompe ; si  $v'$  risque de ne pas respecter les marnages, alors on force le changement de l'état de la pompe (cf. les lignes 3 à 10). Puis le niveau d'eau à la fin de  $t$  est calculé et on ajoute le coût du pompage associé à  $cost$ . Les lignes 11 à 13 ajoutent la pénalité du réservoir non rempli à la fin de la période d'étude.

Cet algorithme fonctionne aussi bien pour la version déterministe que pour la version robuste du RFLATS, grâce à l'utilisation du vecteur  $d$  (prenant par défaut la valeur d'un vecteur nul). Par ailleurs, le choix de notre implémentation permet aussi à cet algorithme de fonctionner pour tous les autres types de marnages tant qu'on donne en entrée un marnage horaire. Par exemple, cet algorithme permet de calculer les coûts montrés dans la [Figure 6.1](#).

### 6.2.3 Graphe de plus long chemin pour la recherche de la solution pire cas

Le pire cas pour un marnage donné correspond à une répartition du budget d'incertitude afin d'augmenter la consommation et/ou d'augmenter le coût du pompage et/ou d'augmenter le coût de la pénalité du réservoir non rempli. On peut déterminer la répartition du budget menant à l'évolution du niveau d'eau de coût le plus élevé, à l'aide d'un algorithme de recherche de chemin de plus grand coût dans un certain graphe séquentiel sans circuit. On peut créer ce graphe dès lors qu'on arrive à discrétiser les états du système (niveaux d'eau, consommation, débit de la pompe, budget d'incertitude), et cela nécessite parfois un prétraitement sur les données de consommations (comme arrondir les volumes d'eau du remplissage, de la demande et de la déviation à l'unité la plus proche). On crée les sommets du graphe en décrivant : un pas de temps  $t \in \{0, \dots, T\}$ , un niveau d'eau  $v \in \{\underline{V}, \dots, \bar{V}\}$ , un état de la pompe  $x \in \{0, 1\}$  et un budget d'incertitude restant  $b \in \{0, \dots, B\}$  ; et tel que les transitions entre deux sommets  $s_1 = (t_1, v_1, x_1, b_1)$  et  $s_2 = (t_2, v_2, x_2, b_2)$  (où  $t_2 = t_1 + 1$ ) prennent en compte le remplissage  $q_1$ , la consommation  $c_1$  et la déviation  $d_1 = b_2 - b_1$  pour correspondre  $v_1$  à  $v_2$  (tel que  $v_2 = v_1 + q_1 - c_1 - d_1$ ).

La [Figure 6.2](#) montre ce à quoi pourrait ressembler une sous partie de ce genre de graphe, entre  $t = 0$  et  $t = 4$ , dans lequel le budget d'incertitude total

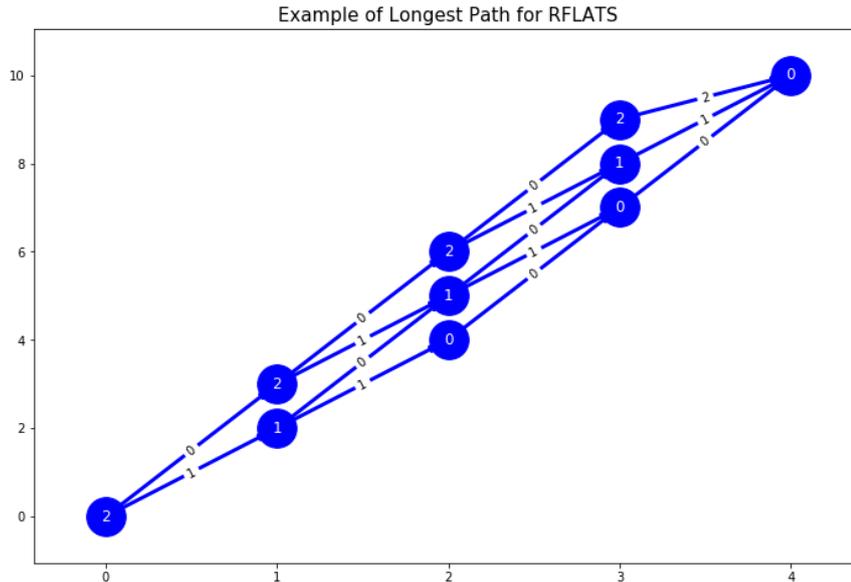


FIGURE 6.2 – Exemple de plus long chemin pour le pire cas de RFLATS

vaut  $B = 2$ . La figure montre des sommets placés aux coordonnées définies par les couples  $(t, v)$ , et portant l'information du budget restant  $b$ ; les transitions montrent l'utilisation du budget (i.e la déviation).

Notons que ce graphe n'a aucun lien avec celui du [Chapitre 5](#) permettant montrer la complexité polynomiale de RFLATS dans le cas déterministe. En effet, dans le précédent graphe les transitions représentent une partition d'un ensemble de solution de marnage, alors que dans ce graphe les transitions représentent l'utilisation du budget d'incertitude.

### 6.2.4 Réduction du graphe de la recherche du pire cas

Le graphe complet de la [Figure 6.2](#) est quand même bien trop grand à créer. On peut heureusement très largement réduire le graphe en utilisant la remarque suivante : certaines répartitions du budget donnent les même planning de pompage et donnent donc des solutions de même coûts, il y a des symétries inutiles dans ce graphe. En effet, l'application d'un marnage correspond un planning d'activation des pompes où des phases de remplissage et de non-remplissage s'enchaînent dans le temps, explicable par une série de pas de temps où l'état de la pompe change. Ainsi, il n'est pas obligatoire de connaître précisément quand est-ce que la déviation est appliquée, mais uniquement la plage de pas de temps où elle est présente. Par exemple, durant une phase de remplissage de  $t_1$  à  $t_2$ ,

un certain montant du budget d'incertitude peut être réparti n'importe comment sans changer le moment où le niveau d'eau dépasse le marnage haut, et donc le planning de pompage. Dans la [Figure 6.2](#), on observe de nombreux chemins de  $(0, 0)$  vers  $(4, 10)$ , et chacun de ses chemins ont le même coût car les phases de pompage sont identiques.

La réduction du graphe diminue le nombre de sommets et transitions en perdant de la précision sur l'utilisation du budget d'incertitude sur des plages de pas de temps. Les sommets conservés correspondent uniquement à ceux d'un début et d'une fin de plage de pas de temps où la pompe change d'état. Il y a une petite exception sur les sommets conservés : lors du passage de **P** à **C**, on peut parfois jouer sur le niveau d'eau avec des déviations pour arrêter la pompe ou non, il faut donc aussi conserver ces sommets là. Dans cette situation précise, si on utilise la déviation avant la fin de **P** ou après la fin de **P**, alors la situation devient totalement différente, cette remarque est illustrée dans la [Figure 6.3](#) où les déviations sont représentées par des symboles "+". Dans la première sous-figure, on observe qu'avec les déviations, on augmente la consommation, ainsi on empêche la pompe de s'arrêter à  $t = 63$ ; puis dans la seconde sous-figure, on observe que si un des "+" est déplacée de  $t = 63$  vers  $t = 65$ , alors la pompe va devoir s'arrêter à  $t = 63$ .

Dans la suite, on l'appellera ce graphe REDUCEDGRAPH et on recherchera son plus long chemin. Le graphe REDUCEDGRAPH peut être créé de la manière suivante : pour le sommet initial et chacun des autres sommets  $s1 = (t1, v1, x1, b1)$ , on va créer jusqu'à  $b1 + 1$  nouveaux sommets  $s2 = (t2, v2, x2, b2)$  et leurs transitions. On peut utiliser un code très similaire à celui de EVAL afin de déterminer quand les marnages vont s'activer, et donc trouver  $t2$ . Plus précisément, durant une phase de non-remplissage, on cherche le moment où  $v_{t2} \geq \underline{v} > v_{t2+1}$ , alors que durant une phase de remplissage, on cherche le moment où  $v_{t2} \leq \bar{v} < v_{t2+1}$ .

L'[Algorithme 6.3](#) est un exemple d'implémentation de la création de REDUCEDGRAPH. La ligne 2 initialise le vecteur des niveaux d'eau "horaires", et la ligne 3 initialise le sommet initial représentant l'état du réservoir à  $t = 0$  (i.e. le réservoir est rempli, la pompe est éteinte et le budget d'incertitude vaut  $B$ ). Puis pour chacun des sommets non terminaux non exploré  $s1$ , on va créer et ajouter les sommets et transition nécessaire (cf. les lignes 4 à 28). Pour  $s1$ , on trouve le moment  $t2$  où les marnages sont censés s'activer (cf. ligne 7 à ligne 12). Puis si  $t2 + 1 = T$ , alors on crée un sommet terminal en utilisant la plus grande quantité de déviations possibles, c'est à dire la valeur minimale entre le manque d'eau ( $v_T - \underline{v}$  et

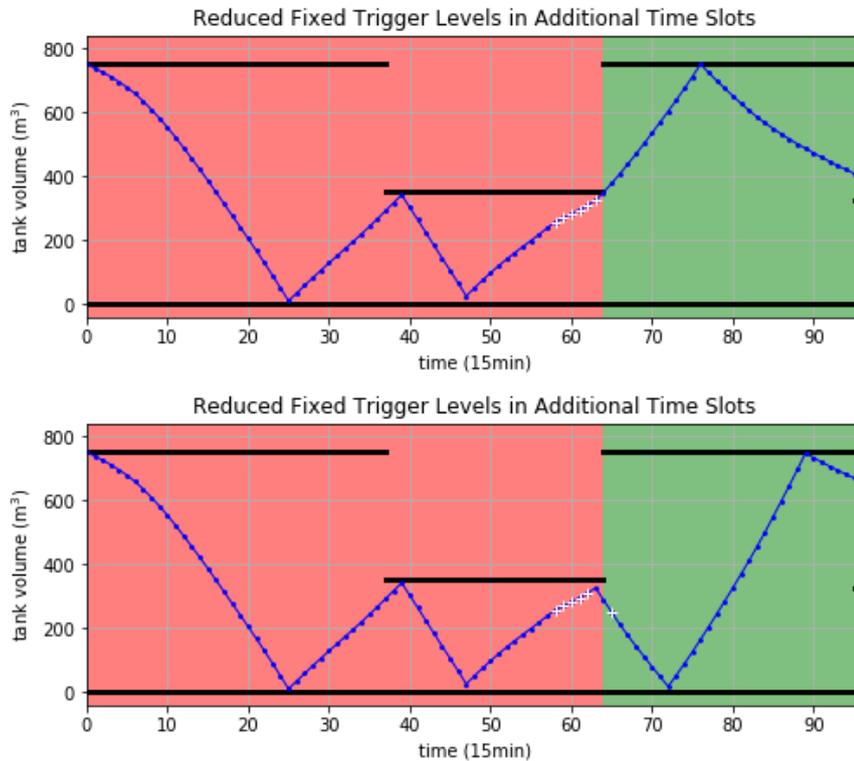


FIGURE 6.3 – Importance de la déviation à  $t = T_P$

la déviation maximale  $(T - t_1) * r\sigma$  (cf. les lignes 13 à 16). Et si  $t_2 + 1 \neq T$ , alors on crée les  $b_1 + 1$  sommets et transitions vers  $s_2 = (t_2, v_2 - b, -x_1, b_1 - b)$  (cf. les lignes 17 à 27). Notons qu'il faut mettre à jour  $t_2$ , si la déviation  $b$  crée une activation des marnages (cf. les lignes 19 à 23).

### 6.2.5 Simplification du graphe de la recherche du pire cas

La méthode avec REDUCEDGRAPH est intéressante, mais les expérimentations montreront qu'elle ne permet pas de trouver la solution robuste dans la limite des 8 heures qu'on s'autorise. Afin de trouver une solution intéressante dans le temps imparti, on s'oriente vers une approche heuristique. Ainsi, on propose une simplification du graphe REDUCEDGRAPH en diminuant le nombre de transitions sortantes d'un sommet.

Pour REDUCEDGRAPH, à partir d'un sommet  $s_1 = (t_1, v_{t_1}, x_1, b_1)$ , on peut retrouver jusqu'à  $b_1 + 1$  transitions sortantes. Pour la simplification, nous allons diminuer ce nombre de transitions sortantes, i.e jouer sur la possibilité d'utilisation du budget d'incertitude. Soit  $s_1 = (t_1, v_{t_1}, x_1, b_1)$  et  $s_2 = (t_2, v_{t_2}, x_2, b_2)$  tel

---

**Algorithme 6.3** Pseudo code pour la création de REDUCEDGRAPH

---

```

1: fonction  $G \leftarrow \text{CREATEREDUCEDGRAPH}(\bar{v}, \underline{v}, B)$ 
2:    $v \leftarrow (\bar{v}, 0, \dots, 0)$ 
3:   Créer le sommet initial  $s_0 = (0, \bar{v}, \text{False}, B)$ 
4:   répéter
5:     Trouver  $s_1$ , un sommet non terminal et non exploré
6:      $(t_1, v_1, x_1, B_1) \leftarrow s_1$ 
7:     pour  $t_2$  de  $t_1$  à  $T$  faire
8:        $v_{t_2+1} \leftarrow v_{t_2} + x_1 * Q - D_{t_2}$ 
9:       si  $v_{t_2+1} < \underline{v}$  ou  $\bar{v} < v_{t_2+1}$  alors
10:        Arrêter la boucle
11:      fin si
12:    fin pour
13:    si  $t_2 + 1 == T$  alors
14:       $b \leftarrow \min(v_T - \underline{v}, (T - t_1) * r\sigma)$ 
15:      Créer le sommet terminal  $s_2 = (T, v_T - b, \neg x_1, B_1 - b)$ 
16:      Ajouter la transition de  $s_1$  vers  $s_2$ 
17:    sinon
18:      pour  $b$  de 0 à  $B_1$  faire
19:        si  $x_1$  et  $\bar{v} < v_{t_2+1} - b$  alors
20:           $t_2 \leftarrow t_2 + 1$ 
21:        sinon si  $\neg x_1$  et  $v_{t_2+1} + Q - b < \underline{v}$  alors
22:           $t_2 \leftarrow t_2 - 1$ 
23:        fin si
24:      Ajouter le sommet intermédiaire  $s_2 = (t_2, v_{t_2} - b, \neg x_1, B_1 - b)$ 
25:      Ajouter la transition de  $s_1$  vers  $s_2$ 
26:    fin pour
27:  fin si
28:  jusqu'à tout les sommets non terminaux sont explorés
29:  retourner  $G$ 
30: fin fonction

```

---

qu'il y ait une transition de  $s_1$  vers  $s_2$  n'utilisant pas de déviation dans REDUCEDGRAPH, ainsi l'utilisation des déviations (positives) peut raccourcir la durée de la plage  $(t_1, t_2)$  si la pompe est éteinte (i.e. on consomme plus) et rallonger la durée de la plage si la pompe est active (i.e. on remplit moins vite), on note les nouvelles fins de plage possible  $t_2' = t_2 \pm t'$ . On remarque que plusieurs montants du budget d'incertitude peuvent aussi donner le même planning de pompage, i.e. correspondant au même couple  $(t_1, t_2')$ . Par exemple, durant une phase de non pompage si  $v_{t_2-1} = 30$ ,  $D_{t_2-1} = 20$  et  $v_{t_2} = 10$  (rappel,  $v_{t_2} = v_{t_2-1} - D_{t_2-1} - d_{t_2-1}$ ), alors on peut utiliser n'importe quel montant du budget  $d_{t_2-1}$  entre 0 et 10 afin

que la plage se termine à  $t_2$ , sinon on peut utiliser n'importe quel montant du budget entre 11 et 30 afin que la plage se termine à  $t_2 - 1$ .

Notons  $t''$  le nombre de pas de temps différents auxquelles appartiennent les  $b+1$  sommets créés par REDUCEDGRAPH à partir d'un sommet  $s_1 = (t_1, v_{t_1}, x_1, b_1)$ . La simplification qu'on propose conserve uniquement deux transitions pour chaque  $t_2'$ . Plus précisément, pour un sommet  $s_1$  on va créer uniquement 2 transitions par  $t_2'$ , soit celle utilisant le plus de budget possible et celle utilisant le moins de budget possible i.e. finissant vers des sommets  $s_2$  où  $t = t_2'$ . Finalement, on crée un total de  $2 * t''$  transitions à partir de  $s_1$ . Dans l'exemple, pour une fin en  $t_2'$  on conserve la transition utilisant 0 et la transition utilisant 10 du budget, et pour une fin en  $t_2' - 1$  on conserve la transition utilisant 11 et la transition utilisant 30; ainsi on réduit donc le nombre de transitions à 4 au lieu de  $B_1$ .

Dans la suite, on appellera ce graphe réduit : SIMPLIFIEDGRAPH. Nos travaux de recherche révèlent que le plus long chemin de SIMPLIFIEDGRAPH n'est pas forcément un des plus long chemin de REDUCEDGRAPH. Bien que SIMPLIFIEDGRAPH ne permet pas de trouver le pire cas exact, nous exposerons aussi des applications afin de montrer ses avantages et de faire des comparaisons avec REDUCEDGRAPH.

### 6.2.6 Algorithme pour la solution robuste

Nous résolvons le problème robuste (6.1-6.4, 5.4-5.8, 5.9''-5.12'', 5.13-5.17) avec l'énumération des marnages de type RFLATS, et pour chacun des marnages on détermine son pire cas avec REDUCEDGRAPH ou SIMPLIFIEDGRAPH.

L'Algorithme 6.4 est un exemple d'implémentation d'algorithme permettant la résolution du problème robuste. Elle prend en entrée *dist\_min* et *step* afin de paramétrer la recherche, puis retourne la solution robuste, son coût et les déviations correspondantes. La ligne 2 de l'algorithme initialise des variables pour définir la meilleure solution (la moins chère). Ensuite il y a 4 boucles *for* pour l'énumération des 4 variables de décision ( $l_{P'}$ ,  $l_{C'}$ ,  $\bar{v}_{P'}$ ,  $\underline{v}_{C'}$ ) (cf. les lignes 3 à 6). Pour chaque combinaison des variables, on récupère le marnage horaire (cf. ligne 7), et on évalue son coût en calculant son pire cas avec REDUCEDGRAPH (ou SIMPLIFIEDGRAPH) (cf. ligne 8). Si la solution est plus intéressante que la meilleure qu'on connaît, alors on la conserve (cf. ligne 9 à 10). Puis on retourne la meilleure solution trouvée, ainsi que son coût et les déviations associées.

Dans la section suivante, nous rapporterons des résultats algorithmiques pour

---

**Algorithme 6.4** Fonction d'énumération des marnages pour la résolution du problème robuste

---

```

1: fonction  $\bar{v}, \underline{v} \leftarrow \text{ENUMRFLATS}(dist\_min, step)$ 
2:    $best\_sol, best\_cost, best\_d \leftarrow \infty, (),$ 
3:   pour  $\bar{v}_{\mathbf{P}'}$  de  $dist\_min$  à  $\bar{V}$ , pas :  $step$  faire
4:     pour  $l_{\mathbf{P}'}$  de 0 à 32 faire
5:       pour  $\underline{v}_{\mathbf{C}'}$  de  $dist\_min$  à  $\bar{V}$ , pas :  $step$  faire
6:         pour  $l_{\mathbf{C}'}$  de 0 à 16 faire
7:            $\bar{v}, \underline{v} \leftarrow \text{RETRIEVERFLATS}(l_{\mathbf{P}'}, l_{\mathbf{C}'}, \bar{v}_{\mathbf{P}'}, \underline{v}_{\mathbf{C}'})$ 
8:            $new\_cost, d \leftarrow \text{REDUCEDGRAPH}(\bar{v}, \underline{v})$ 
9:           si  $new\_cost < best\_cost$  alors
10:             $best\_sol, best\_cost, best\_d \leftarrow new\_cost, (l_{\mathbf{P}'}, l_{\mathbf{C}'}, \bar{v}_{\mathbf{P}'},$ 
11:               $\underline{v}_{\mathbf{C}'})$ ,  $d$ 
12:            fin si
13:          fin pour
14:        fin pour
15:      fin pour
16:    retourner  $best\_sol, best\_cost, best\_d$ 
17: fin fonction

```

---

la recherche du pire cas et pour la résolution du problème robuste sur une instance d'une journée, puis pour la résolution du problème robuste sur une instance d'une semaine. Les différentes courbes des figures montrant l'évolution du niveau d'eau sont obtenues en utilisant [Algorithme 6.1](#) et [Algorithme 6.2](#), e.g. avec  $\bar{v}, \underline{v} \leftarrow \text{RETRIEVERFLATS}(best\_sol)$  et  $best\_cost, best\_v \leftarrow \text{EVAL}(\bar{v}, \underline{v}, 0, \bar{V}, 0, T, best\_d)$ .

## 6.3 Résultats expérimentaux

### 6.3.1 Instance de petit SDE rural

Dans nos expériences, nous considérons la même configuration que dans le [Chapitre 5](#) (cf. [Sous-section 5.5.1](#)). Pour rappel, l'instance considérée comporte un château d'eau d'une capacité de  $\bar{V} = 750m^3$  où le consommation journalière moyenne est d'environ  $3000m^3$ , et la station de pompage est équipée d'une pompe qui peut délivrer  $260m^3$ /heure i.e.  $Q = 65m^3$  sur des pas de temps de 15 minutes. On considère  $\underline{V} = 0m^3$ .

La décomposition STL donne des écart-types  $\sigma$  de  $3.5m^3$ . En prenant  $r = 3$ ,

Solution	Méthode	Coût	CPU
190, 550, 32, 16	SANSINCERTITUDES	19.415663	-
	REDUCEDGRAPH	21.69159	4246ms
	SIMPLIFIEDGRAPH	21.69159	1232ms
395, 407, 13, 11	SANSINCERTITUDES	19.422618	-
	REDUCEDGRAPH	21.903623	1344ms
	SIMPLIFIEDGRAPH	21.899053	58ms
700, 50, 4, 4	SANSINCERTITUDES	19.928318	-
	REDUCEDGRAPH	23.800053	1904ms
	SIMPLIFIEDGRAPH	23.800053	159ms

TABLEAU 6.1 – Résultats expérimentaux pour la résolution du pire cas sur 24 heures

les déviations respectent  $0 \leq d_t \leq \lfloor r * \sigma \rfloor = 10m^3$ . Le budget d'incertitude est fixé à  $B = \lfloor 3 * \sqrt{T} * r^2 * \sigma \rfloor = 300m^3$ , soit un dixième de la consommation journalière.

### 6.3.2 Résultats numériques pour la recherche du pire cas

Dans cette sous-section, nous montrons les résultats expérimentaux pour la recherche du pire cas pour un marnage donné (i.e. un sous-problème de maximisation) en utilisant différentes méthodes. La première méthode est SANSINCERTITUDES équivalent à la résolution du problème déterministe ; la deuxième méthode est REDUCEDGRAPH équivalent à la résolution du problème robuste ; et la dernière méthode est SIMPLIFIEDGRAPH équivalent à une résolution approchée du problème robuste. Notons que SIMPLIFIEDGRAPH ne trouve pas la solution robuste que dans peu de situations, mais elle a l'avantage d'être largement plus rapide que REDUCEDGRAPH.

Le [Tableau 6.1](#) montre des résultats pour l'évaluation du pire cas d'un marnage serré (190, 550, 32, 16), un certain marnage neutre (395, 407, 13, 11) et un marnage lâche (700, 50, 4, 4). La colonne intitulée 'Solution' indique la solution (i.e. les quatre variables de décision  $\bar{v}_{P'}$ ,  $\underline{v}_{C'}$ ,  $l_{P'}$ ,  $l_{C'}$ ) ; la colonne intitulée 'Méthode' indique la méthode de résolution parmi SANSINCERTITUDES, REDUCEDGRAPH et SIMPLIFIEDGRAPH ; la colonne intitulée 'Coût' indique le coût de la solution ; et la colonne intitulée 'CPU' indique le temps cpu total résultant.

Le [Tableau 6.2](#) montre les tailles (en terme de nombre de sommets et arêtes) des graphes générés par les méthodes REDUCEDGRAPH et SIMPLIFIEDGRAPH, où on fait varier uniquement  $\bar{v}_{P'}$ , les trois autres variables sont fixées :  $\underline{v}_{C'} = 554$ ,

$\bar{v}_{\mathbf{P}'}$	REDUCEDGRAPH	SIMPLIFIEDGRAPH
130	(12208, 713559)	(6719, 32849)
175	(11799, 773126)	(4793, 19660)
220	(11356, 727286)	(3269, 12556)
265	(11865, 810864)	(3100, 10778)
310	(11726, 849469)	(3283, 11312)
355	(12088, 723689)	(2539, 8461)
400	(10177, 573596)	(1782, 4728)
445	(8887, 576727)	(1557, 3814)
490	(8210, 561423)	(1656, 4007)
535	(8338, 549664)	(1584, 3783)
580	(8431, 539599)	(1729, 4104)
625	(8895, 534072)	(1639, 4063)
670	(9547, 545673)	(1583, 4106)
715	(9143, 476465)	(1392, 3616)
750	(8820, 420071)	(1328, 3251)

TABLEAU 6.2 – Comparatif des tailles de graphe de la solution  $(\bar{v}_{\mathbf{P}'}, 554, 30, 4)$  pour les méthodes REDUCEDGRAPH et SIMPLIFIEDGRAPH

$l_{\mathbf{P}'} = 30$ ,  $l_{\mathbf{C}'} = 4$ . La colonne intitulée ' $\bar{v}_{\mathbf{P}'}$ ' indique la valeur de  $\bar{v}_{\mathbf{P}'}$ ; les colonnes intitulées 'REDUCEDGRAPH' et 'SIMPLIFIEDGRAPH' montre la taille des graphes en question.

Les Figure 6.4, Figure 6.5 et Figure 6.6 montrent respectivement l'évolution du volume d'eau horaire pour le marnage serré, le marnage neutre et le marnage lâche pour différentes méthodes parmi SANSINCERTITUDES, REDUCEDGRAPH et SIMPLIFIEDGRAPH. La courbe et les points bleus représentent le volume d'eau dans le château d'eau (en  $m^3$ ). Les symboles blancs "+" représentent les pas de temps où on applique une déviation positive non-nulle. Les lignes noires représentent les marnages (en  $m^3$ ). La zone en rouge représente les  $\mathbf{P}$ , tandis que la zone en vert représente les  $\mathbf{C}$ .

Les résultats présentés dans les Tableau 6.1 et Tableau 6.2, et dans les Figure 6.4, Figure 6.5 et Figure 6.6 suggèrent les commentaires suivants :

- La solution pire-cas utilise bien le budget d'incertitude pour décaler les moments de pompage (e.g. dans la Figure 6.6, les déviations avancent et allongent la phase de pompage prévue de  $t = 25$  à  $t = 56$ ), et entraîne parfois de grand changement dans l'évolution du niveau d'eau (e.g. dans la Figure 6.4, il y a un cycle de remplissage qui "disparaît").
- En regardant le niveau d'eau à la fin de la période ( $T = 96$ ), on observe

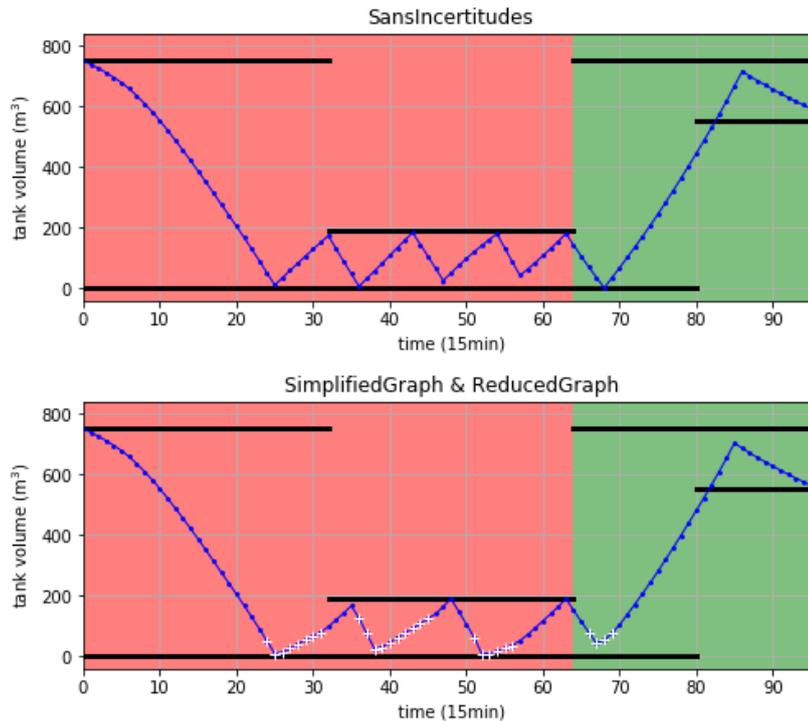


FIGURE 6.4 – Évolution du niveau pour un marnage serré sur 24 heures

que le marnage serré est plus robuste que le marnage neutre ou le marnage lâche (comme mentionné dans la [Sous-section 5.2.4](#)).

- En regardant le [Tableau 6.1](#), la présence des déviations (et de la résolution du pire cas) augmente le coût du marnage serré de  $21.69159 - 19.415663 = 2.275927$ , le coût du marnage neutre de  $21.903623 - 19.422618 = 2.481005$ , et l'augmentation est plus forte pour le marnage lâche ( $23.800053 - 19.928318 = 3.871735$ ). Ces augmentations sont créées d'une part par l'augmentation de la consommation et d'autre part par la résolution du pire cas pour la solution. Notons que pompé  $Bm^3$  d'eau (i.e. l'augmentation de la consommation) durant  $\mathbf{P}$  coûterait environ 2.1324 et coûterait environ 1.5231 durant  $\mathbf{C}$ .
- De manière générale, la majorité du budget d'incertitude est utilisée durant  $\mathbf{P}$ .
- Dans le [Tableau 6.1](#), on mesure aussi des écarts de coût entre les solutions de REDUCEDGRAPH et SIMPLIFIEDGRAPH pour le marnage neutre. On observe cette différence lorsque la solution utilise des déviations que SIMPLIFIEDGRAPH ne peut pas fournir durant le passage de  $\mathbf{P}$  à  $\mathbf{P}'$  (cf. la

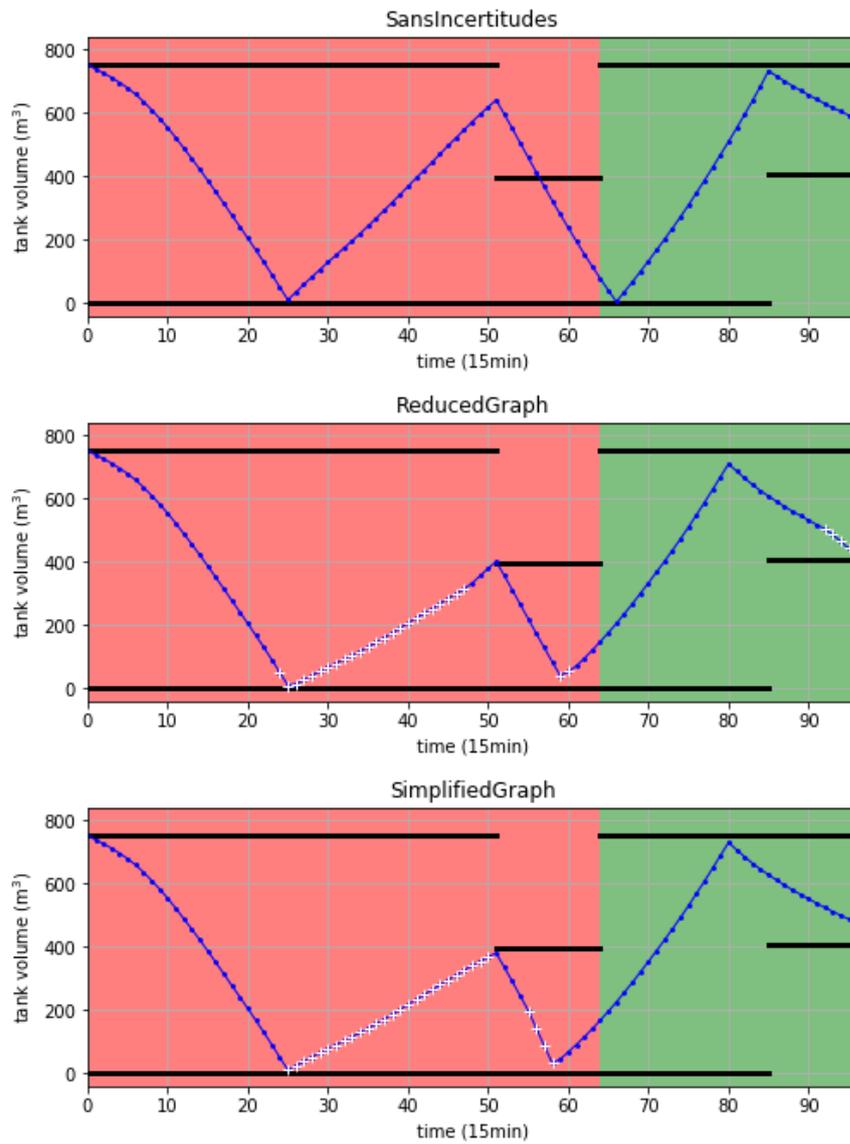


FIGURE 6.5 – Évolution du niveau pour un marnage neutre sur 24 heures

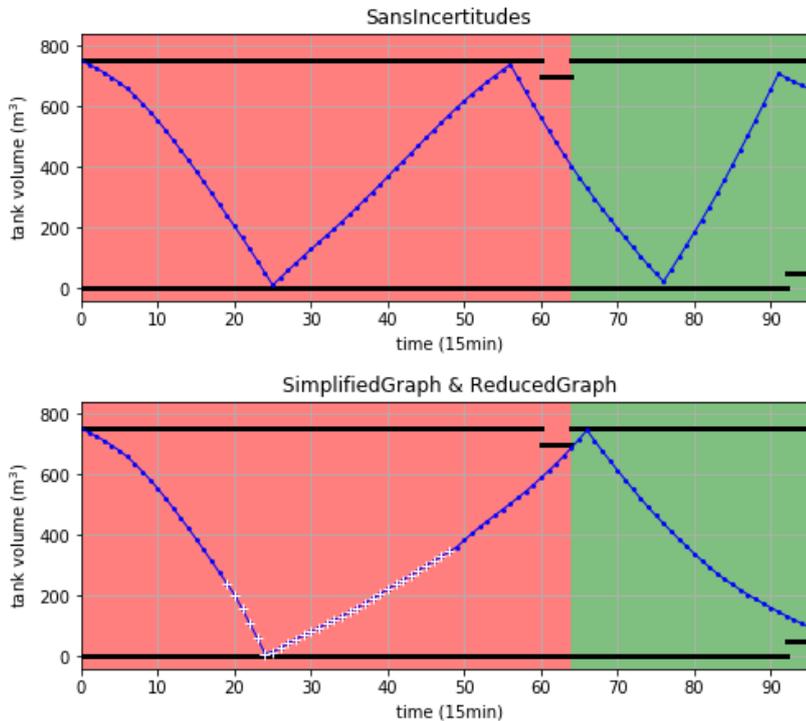


FIGURE 6.6 – Évolution du niveau pour un marnage lâche sur 24 heures

deuxième et troisième sous-figure de Figure 6.5). Cette situation existe mais est assez rare, car on peut souvent fournir la quantité de déviation suffisante durant des transitions antérieures sans modifier les moments de pompage. L'écart de coût est assez faible, et correspond à un pompage commençant ou terminant un pas de temps plus tôt ou plus tard (soit  $Q * 0.0001 = 0.0065$ , tel que 0.001 est la différence de prix maximale au sein de  $\mathbf{P}$ ).

- D'après le Tableau 6.2, on remarque que pour la solution  $(\bar{v}_{\mathbf{P}'}, 554, 30, 4)$ , la taille du graphe est plus grande pour REDUCEDGRAPH que pour SIMPLIFIEDGRAPH avec 2 à 5 fois plus de sommets et 20 à 130 fois plus de transition. Le Tableau 6.1 révèle que la méthode approchée SIMPLIFIEDGRAPH est largement plus rapide que REDUCEDGRAPH (entre 3 et 23 fois plus rapide). Remarque, en moyenne, chaque sommet possèdent 50 arrêtes sortantes pour REDUCEDGRAPH, alors qu'ils n'en ont que 4 pour SIMPLIFIEDGRAPH.

Des tests sur 115200 évaluations montrent que le temps moyen pour la résolution du pire-cas avec REDUCEDGRAPH est de 2312ms et avec SIMPLIFIEDGRAPH est de 225ms.

Solution	Méthode	Coût	CPU
Marnage de référence	SANSINCERTITUDES	20.619723	-
	ENUMRFLATS_R	22.823038	-
	ENUMRFLATS_S	22.823038	-
Solution déterministe 215, 616, 31, 4	SANSINCERTITUDES	18.961118	262ms
	ENUMRFLATS_R	21.566328	-
	ENUMRFLATS_S	21.566328	-
Solution robuste par ENUMRFLATS_S 130, 620, 22, 7	SANSINCERTITUDES	19.210198	-
	ENUMRFLATS_R	-	out of time
	ENUMRFLATS_S	21.386343	7h13min

TABLEAU 6.3 – Coût de la solution de référence, la solution déterministe et la solution robuste sur l’instance de 24 heures

### 6.3.3 Résultats expérimentaux par résolution du problème robuste

Dans cette sous-section, nous montrons les résultats pour la résolution du problème robuste (i.e. le problème principal). Pour résoudre le problème robuste on appelle ENUMRFLATS pour effectuer l’énumération avec un pas de recherche *step*. Pour rappel, le coeur de la résolution du pire-cas se base sur le décalage des moments de pompage (i.e. des indices de temps), ainsi dans nos expériences nous avons choisi de fixer  $step = 45m^3$ . En effet, cette valeur n’est pas inférieure aux demandes de chaque pas de temps (entre 12 et  $45m^3$  cf. [Figure 5.12](#)) et assez proche du débit  $Q$ , ainsi cela permet de jouer sur les indices de temps où la pompe sera active. La résolution du problème robuste avec ENUMRFLATS avec un pas de recherche de  $45m^3$  correspond donc à l’évaluation de 115200 de marnages. On note ENUMRFLATS\_R la variante de ENUMRFLATS utilisant REDUCEDGRAPH et ENUMRFLATS\_S utilisant SIMPLIFIEDGRAPH.

Le [Tableau 6.3](#) montre des résultats pour le marnage de référence, pour le marnage optimal de la version déterministe et pour le marnage optimal de la version robuste suivant différentes méthodes de résolutions. La colonne intitulée ‘Solution’ indique la solution ; la colonne intitulée ‘Méthode’ indique la méthode de résolution parmi SANSINCERTITUDES, ENUMRFLATS\_R et ENUMRFLATS\_S ; la colonne intitulée ‘Coût’ indique le coût de la solution ; et la colonne intitulée ‘CPU’ indique le temps cpu total résultant lorsque la ligne correspond à un résultat d’optimisation.

Les [Figure 6.7](#) et [Figure 6.8](#) montrent respectivement l’évolution du volume

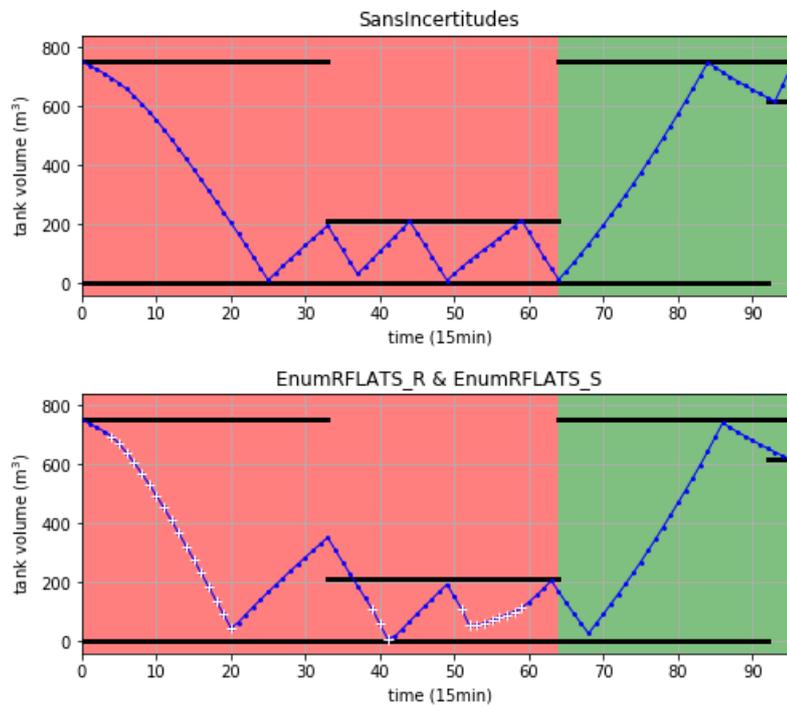


FIGURE 6.7 – Évolution du niveau pour la solution déterministe sur 24 heures

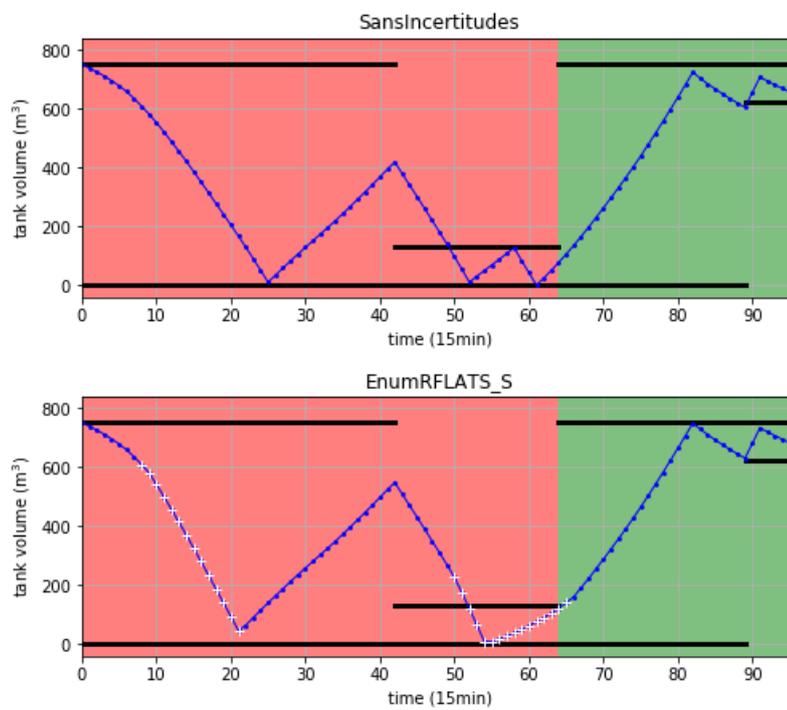


FIGURE 6.8 – Évolution du niveau pour la solution robuste sur 24 heures

d'eau horaire pour les deux marnages optimaux, sans puis avec déviations sur les consommations. La courbe et les points bleus représentent le volume d'eau dans le château d'eau (en  $m^3$ ). Les symboles blancs "+" représentent les pas de temps où on applique une déviation positive non nul. Les lignes noires représentent les marnages (en  $m^3$ ). La zone en rouge représente les **P**, tandis que la zone en vert représente les **C**.

Les résultats présentés dans le [Tableau 6.3](#) et dans les [Figure 6.7](#) et [Figure 6.8](#) suggèrent les commentaires suivants :

- La solution robuste est trouvée en 7 heures et 13 minutes par la méthode ENUMRFLATS\_S. En 8 heures, la méthode ENUMRFLATS\_R n'a exploré que 10% de son espace de recherche.
- La solution déterministe est un marnage serré (cf. [Chapitre 5](#)), alors que la solution robuste est un marnage très serré.
- La présence des déviations augmente le coût pour la solution de référence de  $22.823038 - 20.619723 = 2.203315$ , le coût pour la solution déterministe de  $21.566328 - 18.961118 = 2.60521$ , et le coût pour la solution robuste de  $21.386343 - 19.210198 = 2.176145$ . La solution de référence peut concurrencer la solution robuste, mais elle produit bien trop de changement d'état de la pompe. La solution déterministe est clairement moins intéressante que la solution robuste avec une augmentation 20% supérieure ( $\frac{2.60521-2.176145}{2.176145} = 0.197$ ).
- Les courbes de la [Figure 6.7](#) révèlent que la solution déterministe a un mauvais pire cas, car les déviations permettent au niveau d'être proche du marnage haut à la fin de **P'** au lieu d'être proche du marnage bas.

Même si ENUMRFLATS\_S ne trouve pas la solution optimale, l'écart de coût avec ENUMRFLATS\_R est minime par rapport à l'augmentation créée par les déviations. On recommande l'utilisation de la méthode approchée ENUMRFLATS\_S avec un pas de recherche de  $45m^3$  pour résoudre le problème robuste pour une journée type. Des pas de recherche plus petit risque de ne pas terminer l'exécution durant les 8 heures autorisées alors que des pas de de recherche plus grand n'est plus assez proche de la demande.

Solution	Méthode	Coût
Marnage déterministe reconstruit 6 jours	SANSINCERTITUDES	114.650643
	ENUMRFLATS_S	129.182233
Marnage robuste reconstruit 6 jours	SANSINCERTITUDES	114.831018
	ENUMRFLATS_S	127.212083

TABLEAU 6.4 – Coût de la solution déterministe et la solution robuste sur l'instance de 6 jours

### 6.3.4 Résultats résultats expérimentaux pour le marnage reconstruit et prix de la robustesse

Dans cette sous-section, nous montrons des résultats pouvant interpréter le prix de la robustesse en faisant intervenir les coûts en présence ou non des déviations pour le marnage déterministe reconstruit et le marnage robuste reconstruit. Les marnages reconstruits sont pour le moment uniquement composés de 6 jours, soit cinq "jour de semaine" et un "samedi" (le "dimanche" est pour le moment ignoré). Comme dans le [Chapitre 5](#), on dispose d'au maximum 8 heures pour résoudre le problème robuste hebdomadaire, c'est pourquoi on peut résoudre les 7 problèmes robustes pour chacune des journées types simultanément en utilisant ENUMRFLATS\_S.

Le [Tableau 6.4](#) montre les coûts pour le marnage déterministe reconstruit et pour le marnage robuste reconstruit avec les méthodes SANSINCERTITUDES et ENUMRFLATS\_S. La colonne intitulée 'Solution' indique la solution ; la colonne intitulée 'Méthode' indique la méthode de calcul parmi SANSINCERTITUDES et ENUMRFLATS\_S ; la colonne intitulée 'Coût' indique le coût de la solution.

Les [Figure 6.9](#) et [Figure 6.10](#) montrent respectivement l'évolution du volume d'eau horaire pour les deux marnages reconstruits, sans puis avec déviations sur les consommations. La courbe et les points bleus représentent le volume d'eau dans le château d'eau (en  $m^3$ ). Les symboles blancs "+" représentent les pas de temps où on applique une déviation positive non nul. Les lignes noires représentent les marnages (en  $m^3$ ). La zone en rouge représente les **P** de 6 :00 à 22 :00, tandis que la zone en vert représente les **C** de 22h00 à 6 :00.

Les résultats présentés dans le [Tableau 6.4](#) et dans les [Figure 6.9](#) et [Figure 6.10](#) suggèrent les commentaires suivants :

- Les courbes de la [Figure 6.9](#) révèlent que la solution déterministe reconstruite a un mauvais pire cas. Tous les jours à 22h (c.f. passage de la zone

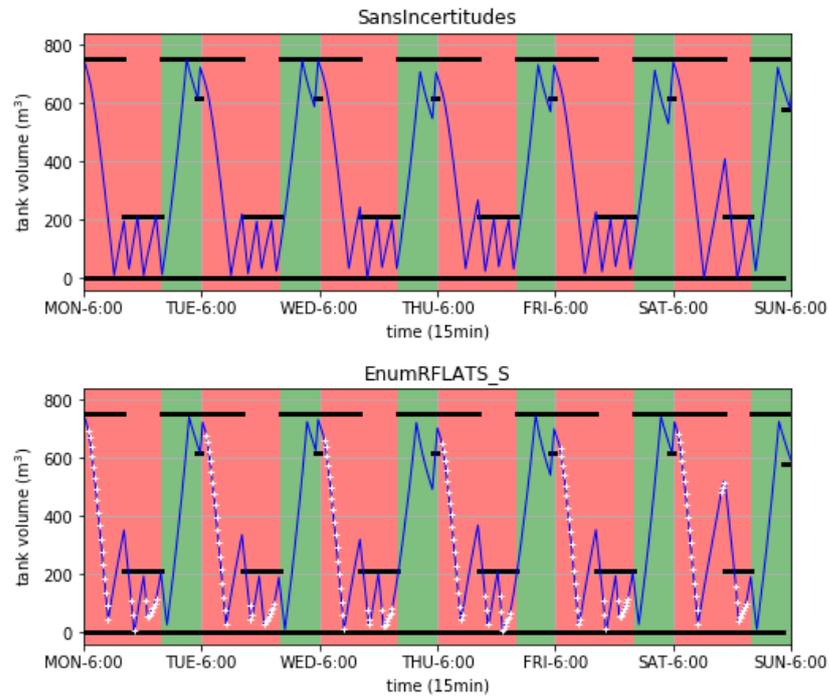


FIGURE 6.9 – Évolution du niveau pour la solution déterministe reconstruit sur 6 jours

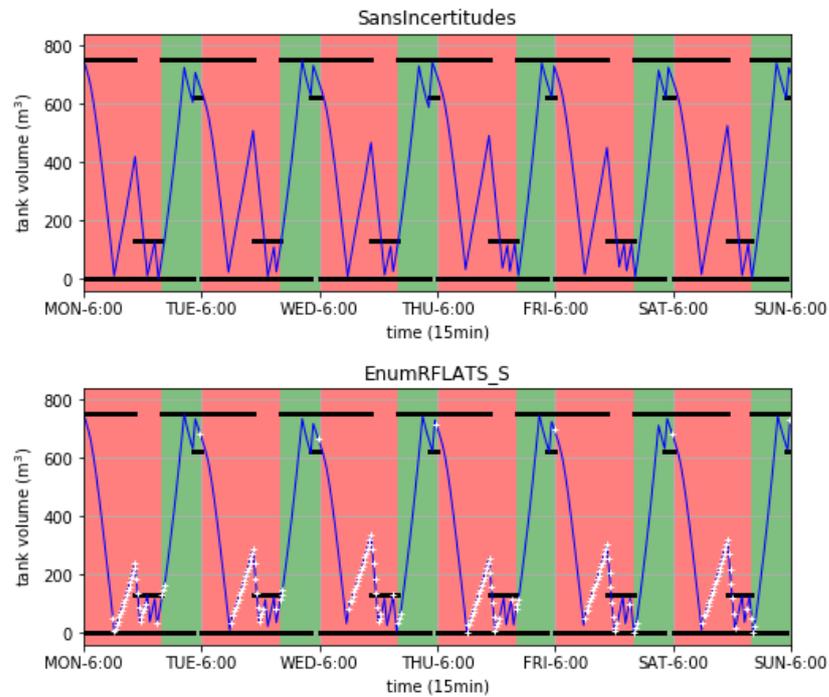


FIGURE 6.10 – Évolution du niveau pour la solution robuste reconstruit sur 6 jours

rouge vers la zone verte), le niveau d'eau est proche du marnage haut ( $200m^3$ ). La présence des déviations augmente le coût pour la solution déterministe reconstruite de  $129.182233 - 114.650643 = 14.53159$ , soit une augmentation journalière moyenne de 2.42193167.

- Les courbes de la [Figure 6.10](#) révèlent que la solution robuste reconstruite a un bon pire cas. Globalement on mesure assez peu d'écarts de niveau d'eau, on note juste quelques changements d'état en plus (le Lundi soir, le Mardi soir, le Jeudi soir et le Samedi soir). La présence des déviations augmente le coût pour la solution déterministe reconstruite de  $127.212083 - 114.831018 = 12.381065$ , soit une augmentation journalière moyenne de 2.06351083 (inférieur au 2.176145 ou 2.20331 vu dans les sous-section précédente).
- La solution déterministe reconstruite est moins intéressante que la solution robuste avec une augmentation 17% supérieure ( $\frac{14.53159-12.381065}{12.381065} = 0.174$ ).
- Notons que pomper  $6 * Bm^3$  d'eau (i.e. le budget d'incertitude des 6 jours) durant **P** coûterait environ 12.364569 et coûterait environ 8.831592 durant **C**. La solution déterministe reconstruite ne contrôle presque pas le niveau d'eau car l'augmentation est bien supérieur au prix d'un pompage naïf durant **P**, i.e. le marnage n'est pas robuste. La solution robuste reconstruite contrôle assez bien le niveau d'eau avec une faible augmentation du coût, l'augmentation est similaire au prix d'un pompage naïf durant **P**, le marnage est bien robuste au déviation (qui augmente le prix suivant le pire cas).
- On peut considérer que la différence  $127.212083 - 114.650643 = 12.56144$  ou 11% ( $\frac{127.212083-114.650643}{114.650643} = 0.11$ ) comme étant le prix de la robustesse.

## 6.4 Résumé du chapitre

Similairement au [Chapitre 4](#), nous proposons une version robuste du problème de gestion de système de distribution d'eau (SDE) en utilisant une combinaison de la décomposition *Seasonal and Trend decomposition using Loess* [31] et un concept d'ensemble d'incertitude similaire à celui de [15]. Le problème consiste donc à trouver les meilleurs marnages de type RFLATS [88] pour une semaine en fonction des demandes d'eaux soumises à différentes sources d'incertitude.

Contrairement au problème du [Chapitre 4](#), la détermination du pire cas n'est pas si simple à cause de la fonction objectif.

Le problème robuste est finalement résolu par énumération des marnages possibles et pour chaque marnage on cherche les plus mauvaises déviations (i.e. son pire cas). Ce pire cas peut être résolu par un algorithme de plus long chemin dans un graphe séquentiel sans circuit, où les sommets représentent les états du système (pas de temps, niveau d'eau, budget d'incertitude restant). On montre aussi une méthode approchée simplifiant le graphe précédent, réduisant fortement le temps nécessaire pour la résolution, et produisant des solutions toutes aussi compétitive en utilisant les 8 heures maximales autorisées. Cette méthode approchée trouve souvent le pire-cas ou retourne des solutions de très bonne qualité de coût très proche du pire-cas. Finalement, les résultats numériques montrent que la solution robuste est un marnage très serré, différent de la solution déterministe, qui en présence des déviations peut être plus mauvaise que la solution robuste de 20% sur les instances de 24 heures et de 17% sur les instances de 6 jours.

CHAPITRE 6. RFLATS AVEC DES INCERTITUDES SUR LES DEMANDES

# Conclusion générale et perspectives

L'objectif de cette thèse est de proposer une solution pour la modélisation et la résolution de modèles d'optimisation pour la gestion de consommation d'énergie multi-flux. Elle présente des contributions scientifiques et des innovations en termes d'applications industrielles autour de l'optimisation de la souscription de contrat et autour de la gestion optimale des réseaux d'eau.

Dans le deuxième chapitre, nous avons modélisé le problème de souscription de contrat d'électricité (PSCE). Nous montrons que le PSCE peut se ramener à un problème de minimisation à flot de coût convexe à objectif séparable, nous montrons aussi qu'il peut se ramener à un problème convexe à objectif séparable sous des contraintes d'ordre total. De nombreux algorithmes efficaces existent pour ces deux classes de problèmes. Cependant, dans le contexte d'application étudié ici, l'effort de calcul nécessaire pour évaluer les termes de la fonction objectif ne peut pas être supposé en  $O(1)$ , mais se repose sur le nombre d'opérations arithmétiques. C'est pourquoi, notre application nécessite une nouvelle analyse des problèmes d'optimisation convexe à objectif séparable sous des contraintes d'ordre total.

Le troisième chapitre met l'accent sur la résolution du PSCE dans sa version déterministe par notre algorithme OPTIM\_SP. Cet algorithme est conçu en se basant sur l'analyse du Chapitre 2, à savoir le remplacement du nombre d'évaluation de fonction par le nombre d'opérations arithmétiques effectuées durant ces évaluations. Cet algorithme est un algorithme itératif, dans lequel on commence à résoudre le problème relâché sans contrainte d'ordre. Puis à chaque itération, on ajoute une nouvelle contrainte et on vérifie si la solution courante respecte la contrainte en question, si besoin on procède à une réévaluation des composantes en question en considérant la contrainte violée comme active dans la solution.

La minimisation de chaque composante se fait par une recherche dichotomique ajustable, où le point évalué est choisi suivant un paramètre ajustable  $a$  et suivant la donnée en entrée, et l'effort de calcul est finalement inférieur à celui d'une dichotomie classique. Sous l'hypothèse où chaque terme de la fonction objectif peut être calculé en  $O(1)$ , on montre que la complexité de `OPTIM_SP` est de  $O(K^2 \log(C))$  alors que les meilleurs algorithmes de la littérature se terminent en  $O(K \log(C))$ . Dans notre application, l'hypothèse précédente n'est pas vérifiée, et les expériences numériques effectuées confirment que pour presque toutes les valeurs de  $a$ , notre procédure de dichotomie ajustable diminue le nombre d'évaluation des termes de la fonction objectif et l'algorithme `OPTIM_SP` est finalement largement plus rapide.

Dans le quatrième chapitre, on propose la modélisation et la résolution d'une version robuste du PSCE. Cette version robuste se base sur l'ensemble d'incertitude construit à partir d'un budget d'incertitude pour les déviations des consommations. On montre que ce problème fait toujours partie de la classe des problèmes convexes à objectif séparables sous des contraintes d'ordre total, ainsi on peut toujours utiliser `OPTIM_SP`. Cependant, la différence est au niveau de l'évaluation de la fonction objectif qui nécessite cette fois la détermination d'un pire-cas se ramenant à la résolution d'un problème de sac à dos à choix multiples de complexité linéaire. La complexité finale est donc toujours polynomiale.

L'algorithme `OPTIM_SP` que nous proposons pour le problème de souscription de contrat d'électricité pourrait ouvrir de perspectives algorithmiques nouvelles dans l'optimisation unidimensionnelle lorsque chaque point est associé avec de très grande volume de données. En effet, le calcul de la complexité (et donc de l'efficacité) d'un très grand nombre d'algorithmes d'optimisation unidimensionnelle jusqu'à présent repose sur l'hypothèse que l'évaluation de la fonction objectif à tout point peut se faire un temps constant ( $O(1)$ ). Nous avons montré que cette hypothèse n'est plus vraiment pertinente pour mesurer l'efficacité des algorithmes lorsque l'évaluation de la fonction objectif à un point donné dépend aussi de la longueur historique des données. Nous avons montré avec `OPTIM_SP` que dans ce cas, une modification d'algorithmes existants portant sur un choix judicieux et dynamique des points à évaluer qui prend en compte l'historique des données pourrait permettre d'une amélioration très nette au niveau du temps de CPU.

Dans le cinquième chapitre, on modélise le problème de la gestion hebdomadaire des réseaux d'eau en utilisant des niveaux de marnage de type Reduced Fixed trigger Levels in Additional Time Slots (RFLATS). Le modèle se compose de contraintes basiques de réseau, de contraintes d'activations de marnage, de contraintes de choix du marnage et de contraintes définissant les périodes artificielles. Les résultats expérimentaux montrent que RFLATS produit un marnage avec un très bon coût sans générer un trop grand nombre de changements d'état de la pompe, de plus ils montrent que RFLATS est aussi plus robuste face à des déviations dans les demandes. Finalement, la robustesse des solutions journalières obtenues nous permet de réduire le problème hebdomadaire en la résolution de sept problèmes journaliers, la solution finale est obtenue en reconstruisant le marnage à partir des sept solutions correspondantes. On se compare à une solution de référence utilisant deux mécanismes afin de se protéger des situations de pannes, la première augmente le nombre de pompes (avec des back-up) et la deuxième conserve un niveau d'eau très élevé (avec un grand nombre de changements d'état de la pompe). Les résultats numériques montrent que RFLATS obtient des résultats meilleurs que la solution de référence en terme de coût et en terme de nombre de changements d'état de la pompe.

Le dernier chapitre propose une version robuste du problème des réseaux d'eau. Nous formulons la version robuste en utilisant la même méthodologie que pour le PSCE. La fonction objectif n'a pas de forme remarquable, ainsi nous proposons une énumération des solutions pour résoudre la version robuste du problème. Pour chaque marnage, il faut déterminer son pire cas, nous montrons que ce pire-cas se ramène à chercher le plus long chemin dans un graphe séquentiel sans circuit, où on modélise les états du réservoir par les sommets du graphe et tel que l'évolution du niveau d'eau se traduit par les transitions. On propose aussi une réduction de ce graphe, permettant de trouver une solution approchée de bonne qualité. Les résultats expérimentaux montrent que la méthode approchée trouve souvent la solution robuste, cependant on remarque qu'elle ne trouve pas les solutions utilisant une quantité de déviation très différente de celles qu'on autorise pour le pas de temps représentant le passage des heures pleines aux heures creuses. En effet, durant ce pas de temps précis, les deux transitions qu'on créees ont de forte chance de ne pas utiliser de déviation et tout le budget restant. La méthode approchée est typiquement 10 fois plus rapide que la méthode utilisant le graphe complet, on recommande ainsi son utilisation pour résoudre ce genre

de problème surtout en présence d'une limite de temps pour la résolution du problème.

En terme de perspectives, on pourra travailler sur un modèle du réseau d'eau avec deux pompes, où chacune des pompes possède ses propres niveaux de marnage. Le modèle devra prendre en compte chacune des contraintes de marnage pour chacune des pompes, mais la difficulté réside dans l'adaptation des contraintes liant le niveau d'eau et l'activation des marnages : [Équation \(5.6\)](#) et [Équation \(5.7\)](#). En effet, si ces contraintes sont prises séparément et que chacune de ces contraintes prévoit l'arrêt de sa pompe correspondante, alors il est possible qu'en réalité l'arrêt des deux pompes ne permet plus de fournir assez d'eau et donc d'atteindre les marnages hauts de ces pompes. Pour surmonter ce problème, il est nécessaire de jouer sur la discrétisation du temps. Une première approche peut être une méthode approchée où on considère une discrétisation du temps plus fine. Dans une autre approche, on peut utiliser le pourcentage d'utilisation de chacune des pompes, ainsi les variables d'activation de pompes ne sont plus binaires mais continues entre 0 et 1. Ce modèle sera ainsi le premier pour le marnage de type RFLATS avec plusieurs pompes.

Un approfondissement de l'étude du marnage robuste optimal peut être intéressant. Dans notre application, les solutions robustes obtenues sont souvent des marnages très serrés durant les périodes additionnelles de RFLATS (i.e. le marnage haut et le marnage bas sont très proches durant ces périodes de temps), car ils permettent de mieux contrôler le niveau d'eau durant les périodes additionnelles lorsqu'il y a des déviations. Dans le but de réduire l'espace de recherche de l'énumération des marnages, il semble intéressant d'étudier la structure des solutions robustes. Dans un premier temps, on peut se placer dans une instance où la consommation est nulle durant les périodes additionnelles, dans ce cas le niveau d'eau va stagner à une valeur fixe durant ces pas de temps. Il semble ainsi nécessaire d'appliquer un marnage infiniment serré afin de permettre le contrôle voulu, ce qui n'est pas possible car on force une distance minimale entre le marnage haut et le marnage bas (cf. [Équation \(5.13\)](#)). Dans cette situation, la solution robuste n'est pas un marnage très serré, et l'utilisation du budget d'incertitude doit respecter deux règles :

- ne pas guider le niveau d'eau au plus haut durant les périodes additionnelles, car la solution serait de mauvaise qualité,
- ne pas guider le niveau d'eau vers une valeur trop basse durant les périodes

additionnelles, car les déviations permettraient l'activation de la pompe, puis le niveau d'eau stagnerait au plus haut.

Il y a une relation entre les paramètres du problème (le marnage, la demande, le budget d'incertitude etc ...) permettant de garantir la robustesse du marnage, et comprendre cela permettrait d'éliminer rapidement certaines solutions de l'espace de recherche.

On peut aussi proposer des travaux complémentaires afin de vérifier expérimentalement la robustesse des solutions dites robustes, en se basant par exemple sur la méthodologie exposée dans les travaux de Minoux, M. (2018) [79]. On peut déterminer des courbes de la 'Value at Risk' pour différents niveaux de risque, qui donne une estimation des pertes qui ne devrait pas être dépassée. On peut par exemple obtenir ces courbes en tirant au hasard un grand nombre de scénarios de déviations possibles, puis en déterminant le coût de la solution optimale suivant les déviations tirées, ensuite en traçant l'histogramme des coûts obtenus. On s'intéresse ici à comparer les différentes courbes de Value at Risk comme niveau de risque accepté ; celles obtenues avec les solutions optimales robustes par rapport à celles obtenues pour la solution optimale déterministe (correspondant aux valeurs moyennes des paramètres incertains).

Enfin, on pourrait étendre l'utilisation de notre méthode de modélisation de problème robuste qui combine les résultats de la *Seasonal and Trend decomposition using Loess* de Cleveland, R. B., Cleveland, W. S., McRae, J.E., et Terpenning, I. (1990) [31] et la construction d'un ensemble d'incertitude du type Bertsimas, D., et Sim, M. (2004) [15]. En effet, dès lors qu'on considère les données sous-forme d'une série temporelle, alors on peut facilement modéliser les problèmes dans une version robuste. Ces problèmes peuvent remplacer la donnée par la valeur moyenne de la décomposition ou par une prédiction de celle-ci ; et les déviations peuvent être expliqués par le résiduel de la décomposition.

## *CONCLUSION*

---

# Bibliographie

- [1] Folorunso Taliha Abiodun and Fatimah Sham Ismail. Pump scheduling optimization model for water supply system using AWGA. In 2013 IEEE Symposium on Computers & Informatics (ISCI), pages 12–17, Langkawi, Malaysia, April 2013. IEEE. 70
- [2] Ravindra K. Ahuja, Dorit S. Hochbaum, and James B. Orlin. Solving the Convex Cost Integer Dual Network Flow Problem. Management Science, 49(7) :950–964, July 2003. 28, 33
- [3] Ravindra K. Ahuja and James B. Orlin. A Fast Scaling Algorithm for Minimizing Separable Convex Functions Subject to Chain Constraints. Operations Research, 49(5) :784–789, October 2001. 13, 29, 33, 35, 42, 49
- [4] Stefano Alvisi and Marco Franchini. A Methodology for Pumping Control Based on Time Variable Trigger Levels. Procedia Engineering, page 8, 2016. 70
- [5] Stefano Alvisi and Marco Franchini. A robust approach based on time variable trigger levels for pump control. Journal of Hydroinformatics, 19(6) :811–822, November 2017. 73
- [6] Stefano Alvisi, Marco Franchini, and Alberto Marinelli. A short-term, pattern-based model for water-demand forecasting. Journal of Hydroinformatics, 9(1) :39–50, January 2007. 70
- [7] Aijun An, Ning Shan, Christine Chan, Nick Cercone, and Wojciech Ziarko. Discovering rules for water demand prediction : An enhanced rough-set approach. Engineering Applications of Artificial Intelligence, 9(6) :645–653, December 1996. 70
- [8] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The Internet of Things : A survey. Computer Networks, 54(15) :2787–2805, October 2010. 4

- [9] Miriam Ayer, H. D. Brunk, G. M. Ewing, W. T. Reid, and Edward Silverman. An Empirical Distribution Function for Sampling with Incomplete Information. The Annals of Mathematical Statistics, 26(4) :641–647, 1955. Publisher : Institute of Mathematical Statistics. [28](#)
- [10] Thibaut Barbier. Modélisation de la consommation électrique à partir de grandes masses de données pour la simulation des alternatives énergétiques du futur. PhD thesis, Université de recherche Paris Sciences et Lettres PSL Research University, December 2017. [5](#)
- [11] A. Ben-Tal and A. Nemirovski. Robust solutions of uncertain linear programs. Operations Research Letters, 25(1) :1–13, August 1999. [11](#)
- [12] Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. Robust optimization, 2008. [11](#)
- [13] Aharon Ben-Tal and Arkadi Nemirovski. Robust solutions of Linear Programming problems contaminated with uncertain data. Mathematical Programming, 88(3) :411–424, September 2000. [11](#)
- [14] J. G. Bene, I. Selek, and Cs. Hős. Comparison of deterministic and heuristic optimization solvers for water network scheduling problems. Water Supply, 13(5) :1367–1376, September 2013. [70](#)
- [15] Dimitris Bertsimas and Melvyn Sim. The Price of Robustness. Operations Research, 52(1) :35–53, February 2004. [12](#), [13](#), [14](#), [51](#), [61](#), [102](#), [124](#), [131](#)
- [16] M. J. Best and R. Y. Tan. An  $O(n^3 \log n)$  strong polynomial algorithm for an isotonic regression knapsack problem. Journal of Optimization Theory and Applications, 79(3) :463–478, December 1993. [28](#)
- [17] Michael J. Best and Nilotpai Chakravarti. Active set algorithms for isotonic regression ; A unifying framework. Mathematical Programming, 47(1-3) :425–439, May 1990. [28](#)
- [18] Michael J. Best, Nilotpai Chakravarti, and Vasant A. Ubhaya. Minimizing Separable Convex Functions Subject to Simple Chain Constraints. SIAM Journal on Optimization, 10(3) :658–672, January 2000. [33](#)
- [19] John R. Birge and François Louveaux. Introduction to Stochastic Programming. Springer Science & Business Media, June 2011. Google-Books-ID : Vp0Bp8kjPxUC. [11](#)
- [20] Lisa J. Blinco, Angus R. Simpson, Martin F. Lambert, and Angela Marchi.

- Comparison of Pumping Regimes for Water Distribution Systems to Minimize Cost and Greenhouse Gases. Journal of Water Resources Planning and Management, 142(6) :04016010, June 2016. 70
- [21] Gratien Bonvin, Sophie Demassey, and Welington de Oliveira. Robust Design of Pumping Stations in Water Distribution Networks. Optimization of Complex Systems : Theory, Models, Algorithms and Applications, 991 :957–967, 2020. 70
- [22] Gratien Bonvin, Sophie Demassey, Claude Le Pape, Nadia Maïzi, Vincent Mazauric, and Alfredo Samperio. A convex mathematical program for pump scheduling in a class of branched water networks. Applied Energy, 185 :1702–1711, January 2017. 70, 81
- [23] Jérémie Bosom. Conception de microservices intelligents pour la supervision de systèmes sociotechniques : application aux systèmes énergétiques. These de doctorat, Université Paris sciences et lettres, December 2020. 2
- [24] Taoufik Bouguera. Capteur communicant autonome en énergie pour l’IoT. These de doctorat, Nantes, March 2019. 5
- [25] H. D. Brunk. Maximum Likelihood Estimates of Monotone Parameters. The Annals of Mathematical Statistics, 26(4) :607–616, 1955. Publisher : Institute of Mathematical Statistics. 28
- [26] G Cembrano, G Wells, J Quevedo, R Pérez, and R Argelaguet. Optimal control of a water distribution network in a supervisory control system. Control Engineering Practice, 8(10) :1177–1188, October 2000. 69
- [27] Cristiano Cervellera, Victoria C.P. Chen, and Aihong Wen. Optimization of a large-scale water reservoir network by stochastic dynamic programming with efficient state space discretization. European Journal of Operational Research, 171(3) :1139–1151, June 2006. 70
- [28] Nilotpai Chakravarti. Isotonic median regression for orders representable by rooted trees. Naval Research Logistics (NRL), 39(5) :599–611, 1992. 28
- [29] Chiung-Yao Chen and Ching-Jong Liao. A linear programming approach to the electricity contract capacity problem. Applied Mathematical Modelling, 35(8) :4077–4082, August 2011. 18
- [30] citepa. <https://www.citepa.org/>. 3

- [31] Robert B. Cleveland, William S. Cleveland, Jean E. McRae, and Irma Terpenning. STL : a seasonal-trend decomposition. Journal of official statistics, 6(1) :3–73, 1990. VII, 7, 8, 13, 14, 51, 53, 61, 102, 124, 131
- [32] George B. Dantzig. Linear Programming under Uncertainty. Management Science, 1(3-4) :197–206, April 1955. Publisher : INFORMS. 11
- [33] Francesco De Angelis, Matteo Boaro, Danilo Fuselli, Stefano Squartini, Francesco Piazza, and Qinglai Wei. Optimal Home Energy Management Under Dynamic Electrical and Thermal Constraints. IEEE Transactions on Industrial Informatics, 9(3) :1518–1527, August 2013. 4
- [34] Ministère de l'Écologie et du Développement durable. <https://www.gouvernement.fr/action/la-cop-21>. 2
- [35] Jean-Baptiste Durand, Laurent Bozzi, Gilles Celeux, and Christian Derquenue. Analyse de courbes de consommation électrique par chaines de Markov cachées. Revue de statistique appliquée, 2003. 5
- [36] Claudia D'Ambrosio, Andrea Lodi, Sven Wiese, and Cristiana Bragalli. Mathematical programming techniques in water network optimization. European Journal of Operational Research, 243(3) :774–788, June 2015. 70
- [37] Bradley J. Eck and Martin Mevissen. Valve placement in water networks : Mixed-integer non-linear optimization with quadratic pipe friction. Report No RC25307 (IRE1209-014), IBM Research, 2012. 70
- [38] Laurent El Ghaoui and Hervé Lebret. Robust Solutions to Least-Squares Problems with Uncertain Data. SIAM Journal on Matrix Analysis and Applications, 18(4) :1035–1064, October 1997. 11
- [39] Laurent El Ghaoui, Francois Oustry, and Hervé Lebret. Robust Solutions to Uncertain Semidefinite Programs. SIAM Journal on Optimization, 9(1) :33–52, January 1998. 11
- [40] Energisme. <https://energisme.com/>. 2
- [41] Dave Evans. The internet of things : How the next evolution of the internet is changing everything. CISCO white paper, 1(2011) :1–11, 2011. 4
- [42] Xi Fang, Satyajayant Misra, Guoliang Xue, and Dejun Yang. Smart grid – the new and improved power grid : A survey. Ieee Communications Surveys and Tutorials, 2011. 3

- 
- [43] Corinna Fischer. Feedback on household electricity consumption : a tool for saving energy? Energy Efficiency, 1(1) :79–104, February 2008. 4
- [44] Gabriele Freni, Mauro De Marchis, and Enrico Napoli. Implementation of pressure reduction valves in a dynamic water distribution numerical model to control the inequality in water supply. Journal of Hydroinformatics, 16(1) :207–217, January 2014. 70
- [45] Mario E. Castro Gama, Quan Pan, Salman M. A. Salman, and Andreja Jonoski. Multivariate optimization to decrease total energy consumption in the water supply system of abbiategrasso (milan, italy). Environmental Engineering and Management Journal, 14(9) :2019–2029, September 2015. Number : 9. 70
- [46] Michael R. Garey and David S. Johnson. Computers and intractability : a guide to the theory of NP-completeness. A series of books in the mathematical sciences. Freeman, New York [u.a], 27. print edition, 1979. 10
- [47] Frank W. Geels. Technological transitions as evolutionary reconfiguration processes : a multi-level perspective and a case-study. Research Policy, 31(8) :1257–1274, December 2002. 2
- [48] Björn Geißler, Oliver Kolb, Jens Lang, Günter Leugering, Alexander Martin, and Antonio Morsi. Mixed integer linear models for the optimization of dynamical transport networks. Mathematical Methods of Operations Research, 73(3) :339–362, June 2011. 70
- [49] Linas Gelazanskas and Kelum A. A. Gamage. Demand side management in smart grid : A review and proposals for future direction. Sustainable Cities and Society, 11 :22–30, February 2014. 4
- [50] Giorgio Graditi, Maria Luisa Di Silvestre, Roberto Gallea, and Eleonora Riva Sanseverino. Heuristic-Based Shiftable Loads Optimal Management in Smart Micro-Grids. IEEE Transactions on Industrial Informatics, 11(1) :271–280, February 2015. 4
- [51] David Hadka and Patrick Reed. Borg : An Auto-Adaptive Many-Objective Evolutionary Computing Framework. 77
- [52] Dirk Helbing and Evangelos Pournaras. Society : Build digital democracy. Nature, 527 :33–34, November 2015. 4

- [53] Dorit S. Hochbaum. Complexity and algorithms for convex network optimization and other nonlinear problems. 4OR, 3(3) :171–216, September 2005. 28
- [54] Sung-Pil Hong, Taegyoon Kim, and Subin Lee. A precision pump schedule optimization for the water supply networks with small buffers. Omega, 82 :24–37, January 2019. 70
- [55] Rob Hopkins. The Transition Handbook : from oil dependency to local resilience ; Manuel de transition. De la Dependance au Petrole a la Resilience Locale. ., October 2010. 2
- [56] Mashor Housh and Elad Salomons. Optimal Dynamic Pump Triggers for Cost Saving and Robust Water Distribution System Operations. Journal of Water Resources Planning and Management, 145(2) :04018095, 2018. Publisher : American Society of Civil Engineers. 70, 73
- [57] Harun Or Rashid Howlader, Hidehito Matayoshi, and Tomonobu Senjyu. Distributed generation incorporated with the thermal generation for optimum operation of a smart grid considering forecast error. Energy Conversion and Management, 96 :303–314, May 2015. 4
- [58] Junyan Hu and Parijat Bhowmick. A consensus-based robust secondary voltage and frequency control scheme for islanded microgrids. International Journal of Electrical Power & Energy Systems, 116 :105575, March 2020. 4
- [59] Hafiz Majid Hussain, Nadeem Javaid, Sohail Iqbal, Qadeer Ul Hasan, Khurshed Aurangzeb, and Musaed Alhussein. An Efficient Demand Side Management System with a New Optimized Home Energy Management Controller in Smart Grid. Energies, 11(1) :190, January 2018. 4
- [60] Jong-Ching Hwang, Jung-Chin Chen, J. S. Pan, and Yi-Chao Huang. CSO and PSO to solve optimal contract capacity for high tension customers. In 2009 International Conference on Power Electronics and Drive Systems (PEDS), pages 246–251, November 2009. ISSN : 2164-5264. 18
- [61] Fatih Issi and Orhan Kaplan. The Determination of Load Profiles and Power Consumptions of Home Appliances. Energies, 11(3) :607, March 2018. 5
- [62] Wooyoung Jeon and Jung Youn Mo. The true economic value of supply-side energy storage in the smart grid environment – The case of Korea. Energy Policy, 121 :101–111, October 2018. 4

- 
- [63] W. A. Thompson Jr. The Problem of Negative Estimates of Variance Components. The Annals of Mathematical Statistics, 33(1) :273–289, March 1962. Publisher : Institute of Mathematical Statistics. 28
- [64] Donghwi Jung, Doosun Kang, Mingu Kang, and Byungseop Kim. Real-time pump scheduling for water transmission systems : Case study. KSCE Journal of Civil Engineering, 19(7) :1987–1993, November 2015. 70
- [65] Alexander V. Karzanov and S. Thomas McCormick. Polynomial Methods for Separable Convex Optimization in Unimodular Linear Spaces with Applications. SIAM Journal on Computing, 26(4) :1245–1275, August 1997. 28, 33
- [66] Amir Kavousian, Ram Rajagopal, and Martin Fischer. Determinants of residential electricity consumption : Using smart meter data to examine the effect of climate, building characteristics, appliance stock, and occupants' behavior. Energy, 55 :184–194, June 2013. 5
- [67] Michael Kazantzis, Angus Simpson, David Kwong, and Shyh Tan. A new methodology for optimizing the daily operations of a pumping plant. Energy, 27 :1–10, January 2002. 71
- [68] Muhammad Asghar Khan, Nadeem Javaid, Anzar Mahmood, Zahoor Ali Khan, and Nabil Alrajeh. A generic demand-side management model for smart grid : A generic demand-side management model for smart grid. International Journal of Energy Research, 39(7) :954–964, June 2015. 4
- [69] Tsung-Ying Lee and Chun-Lung Chen. Iteration particle swarm optimization for contract capacities selection of time-of-use rates industrial customers. Energy Conversion and Management, 48(4) :1120–1131, April 2007. 18
- [70] H. W. Lenstra. Integer Programming with a Fixed Number of Variables. Mathematics of Operations Research, 8(4) :538–548, 1983. Publisher : INFORMS. 10
- [71] Yi Liu, Chau Yuen, Shisheng Huang, Naveed Ul Hassan, Xiumin Wang, and Shengli Xie. Peak-to-Average Ratio Constrained Demand-Side Management With Consumer's Preference in Residential Smart Grid. IEEE Journal of Selected Topics in Signal Processing, 8(6) :1084–1097, December 2014. 4

- [72] Thillainathan Logenthiran, Dipti Srinivasan, and Tan Zong Shun. Demand Side Management in Smart Grid Using Heuristic Optimization. IEEE Transactions on Smart Grid, 3(3) :1244–1252, September 2012. 4
- [73] Manuel Lopez-Ibanez. Operational Optimisation of Water Distribution Networks. Water Resources Research, page 220, 2009. 70
- [74] Denise Lund, Carrie MacGillivray, Vernon Turner, and Mario Morales. Worldwide and regional internet of things (iot) 2014–2020 forecast : A virtuous circle of proven value and demand. International Data Corporation (IDC), Tech. Rep, 1(9), 2014. 4
- [75] Angela Marchi, Angus R Simpson, and Martin F Lambert. Optimization of pump operation using rule-based controls in EPANET2 : a new ETTAR toolkit and correction of energy computation. Journal of Water Resources Planning and Management, page 38, 2016. 73
- [76] John McCulloch, Paul McCarthy, Siddeswara Mayura Guru, Wei Peng, Daniel Hugo, and Andrew Terhorst. Wireless sensor network deployment for water use efficiency in irrigation. In Proceedings of the workshop on Real-world wireless sensor networks - REALWSN '08, page 46, Glasgow, Scotland, 2008. ACM Press. 5
- [77] M. Minoux. A polynomial algorithm for minimum quadratic cost flow problems. European Journal of Operational Research, 18(3) :377–387, December 1984. 28, 33
- [78] M. Minoux. Solving integer minimum cost flows with separable convex cost objective polynomially. In Giorgio Gallo and Claudio Sandi, editors, Netflow at Pisa, Mathematical Programming Studies, pages 237–239. Springer, Berlin, Heidelberg, 1986. 28, 33
- [79] Michel Minoux. Robust and stochastic multistage optimisation under Markovian uncertainty with applications to production/inventory problems. International Journal of Production Research, 56(1-2) :565–583, January 2018. Publisher : Taylor & Francis \_eprint : <https://doi.org/10.1080/00207543.2017.1394597>. 11, 131
- [80] Chukwuka Monyei, Serestina Viriri, Aderemi Adewumi, Innocent Davidson, and Daniel Akinyele. A Smart Grid Framework for Optimally Integrating Supply-Side, Demand-Side and Transmission Line Management Systems.

- 
- Energies, 11(5) :1038, May 2018. Number : 5 Publisher : Multidisciplinary Digital Publishing Institute. 4
- [81] Yurii Nesterov and Arkadii Nemirovskii. Interior point polynomial algorithms in convex programming. Mathematical Programming, 1994. 10
- [82] Arnold N’Goran. Contrôle optimal et gestion énergétique d’une station d’énergie autonome par optimisation robuste. These de doctorat, Université Paris sciences et lettres, May 2020. 3
- [83] Pascal Ogor. Une architecture générique pour la supervision sûre à distance de machines de production avec Internet. These de doctorat, Brest, January 2001. 4
- [84] P.M. Pardalos, G.-L. Xue, and L. Yong. Efficient computation of an isotonic median regression. Applied Mathematics Letters, 8(2) :67–70, March 1995. 28
- [85] Michael Paschke, Kelly Spencer, Nick Waniarcha, Angus Simpson, and Toni Widdop. Genetic algorithms for optimising pumping operations. Australian Water Association, January 2001. 71
- [86] Manisa Pipattanasomporn, Murat Kuzlu, Saifur Rahman, and Yonael Teklu. Load Profiles of Selected Major Household Appliances and Their Demand Response Opportunities. IEEE Transactions on Smart Grid, 5(2) :742–750, March 2014. 5
- [87] Simon L. Prescott and Bogumil Ulanicki. Improved Control of Pressure Reducing Valves in Water Distribution Networks. Journal of Hydraulic Engineering, 134(1) :56–65, January 2008. 70
- [88] Claudia Quintiliani and Enrico Creaco. Using Additional Time Slots for Improving Pump Control Optimization Based on Trigger Levels. Water Resources Management, 33(9) :3175–3186, July 2019. 13, 68, 74, 77, 99, 124
- [89] Melek Rodoplu, Taha Arbaoui, and Alice Yalaoui. Energy Contract Optimization for the Single Item Lot Sizing Problem in a Flow-Shop Configuration and Multiple Energy Sources. IFAC-PapersOnLine, 51(11) :1089–1094, January 2018. 18
- [90] Valerio Salerno and Graziella Rabbeni. An Extreme Learning Machine Approach to Effective Energy Disaggregation. Electronics, 7(10) :235, October 2018. 5

- [91] Tariq Samad, Edward Koch, and Petr Stluka. Automated Demand Response for Smart Buildings and Microgrids : The State of the Practice and Research Challenges. Proceedings of the IEEE, 104(4) :726–744, April 2016. 4
- [92] Michael J. Schell and Bahadur Singh. The Reduced Monotonic Regression Method. Journal of the American Statistical Association, 92(437) :128–135, 1997. Publisher : [American Statistical Association, Taylor & Francis, Ltd.]. 28
- [93] P. Sikka, P. Corke, and L. Overs. Wireless sensor devices for animal tracking and control. In 29th Annual IEEE International Conference on Local Computer Networks, pages 446–454, Tampa, FL, USA, 2004. IEEE (Comput. Soc.). 5
- [94] Gyula Simon, Miklós Maróti, Ákos Lédeczi, György Balogh, Branislav Kusy, András Nádas, Gábor Pap, János Sallai, and Ken Frampton. Sensor network-based countersniper system. In Proceedings of the 2nd international conference on Embedded networked sensor systems - SenSys '04, page 1, Baltimore, MD, USA, 2004. ACM Press. 5
- [95] P. Skworcow, D. Paluszczyszyn, and B. Ulanicki. Pump schedules optimisation with pressure aspects in complex large-scale water distribution systems. Drinking Water Engineering and Science, 7(1) :53–62, June 2014. 70
- [96] Petr Stluka, Datta Godbole, and Tariq Samad. Energy management for buildings and microgrids. In IEEE Conference on Decision and Control and European Control Conference, pages 5150–5157, Orlando, FL, USA, December 2011. IEEE. 2, 4
- [97] Quentin F Stout. Fastest Known Isotonic Regression Algorithms. ., 2019. 28
- [98] Hanne Sæle, Bernt A. Bremdal, T. Troset Engan, Vidar Kristoffersen, Jan A. Foosnæs, Tor Erling Nordal, and J. M. Sletner. Subscribed power-testing new power based network tariffs stimulating for demand response. CIREN 2015, Paper, 1085, 2015. 4, 18
- [99] Wooi-Nee Tan, Ming Tao Gan, and Zheng Ling Tan. Optimization models for demand-side and supply-side scheduling in smart grids. In 2016 IEEE

- 
- 16th International Conference on Environment and Electrical Engineering (EEEIC), pages 1–5, June 2016. 4
- [100] Sylvestre Tatsa. Modélisation et prévision de la consommation horaire d'électricité au Québec : Comparaison de méthodes de séries temporelles. *..*, page 75, 2014. 5
- [101] Edouard Toulouse. La sobriété énergétique, une notion disruptive de plus en plus étudiée. *..*, page 12, 2020. 3
- [102] M-T Tsay, W-M Lin, and J-L Lee. Optimal contracts decision of industrial customers. International Journal of Electrical Power & Energy Systems, 23(8) :795–803, November 2001. 18
- [103] Jakobus E. van Zyl, Dragan A. Savic, and Godfrey A. Walters. Operational Optimization of Water Distribution Systems Using a Hybrid Genetic Algorithm. Journal of Water Resources Planning and Management, 130(2) :160–170, March 2004. Publisher : American Society of Civil Engineers. 70
- [104] Gilles Virone and Norbert Noury. Télé-Surveillance Automatique de l'Activité dans un Habitat Intelligent pour la Santé. *..*, January 2002. 5
- [105] David Wu, Viet Hung Nguyen, Michel Minoux, and Hai Tran. An integer programming model for minimizing energy cost in water distribution system using trigger levels with additional time slots. In 2021 RIVF International Conference on Computing and Communication Technologies (RIVF), pages 1–6, August 2021. ISSN : 2162-786X. 13
- [106] David Wu, Viet Hung Nguyen, Michel Minoux, and Hai Tran. Optimal deterministic and robust selection of electricity contracts. Journal of Global Optimization, June 2021. 4, 13
- [107] Hong-Tzer Yang and Pai-Chun Peng. Improved Taguchi method based contract capacity optimization for industrial consumer with self-owned generating units. Energy Conversion and Management, 53(1) :282–290, January 2012. 18
- [108] Eitan Zemel. An  $O(n)$  Algorithm for the Linear Multiple Choice Knapsack Problem and Related Problems. Inf. Process. Lett., 18(3) :123–128, 1984. 52, 56, 63
- [109] Qian Zhang and Yajie Zhao. Supply-Side Load Optimization After Considering Environmental Cost. Polish Journal of Environmental Studies,

- 29(3) :2455–2466, March 2020. Publisher : HARD Publishing s.c. Jerzy Radecki, Hanna Radecka. 4
- [110] Xin Zhou, Da Yan, Tianzhen Hong, and Xiaoxin Ren. Data analysis and stochastic modeling of lighting energy use in large office buildings in China. Energy and Buildings, 86 :275–287, January 2015. 5

**Résumé :** La collecte automatisée à grande échelle des données de consommation énergétique soulève de nouveaux problèmes d'optimisation. La problématique abordée dans cette thèse est de proposer des solutions pour la modélisation et la résolution de modèles d'optimisation pour la gestion de la consommation d'énergie multi-flux. Les données de consommation sont représentées sous forme de série temporelle.

Nous développons dans un premier temps des modèles déterministes et robustes pour l'optimisation de la souscription de contrat d'électricité. Ce problème est formulé comme la minimisation d'une fonction objectif convexe sous contraintes de structure spéciale (contraintes d'ordre); il est résolu par

un nouvel algorithme efficace qui réduit considérablement l'effort de calcul, en présence d'une grande quantité de données, nécessaire pour évaluer la fonction objectif.

Nous présentons ensuite des modèles déterministes et robustes autour de la gestion optimale des réseaux d'eau en utilisant une stratégie automatisée basée sur l'utilisation des niveaux de marnages. Nous montrons également que le problème peut être réduit en problèmes journaliers plus petits sans perdre trop de précision.

Des résultats numériques sont présentés et révèlent la possibilité de fortes économies dans les différents systèmes énergétiques étudiés.

**Mots-clés :** *Science des données, Optimisation énergétique, Programmation mathématique, Optimisation robuste, Incertitudes.*

**Abstract :** The large-scale automated collection of energy consumption data raises new optimization problems. The issue addressed in this thesis is to propose solutions for modeling and solving optimization models for multi-stream energy consumption management. The consumption data are represented as a time series.

We first develop deterministic and robust models for the optimization of electricity contracting. This problem is formulated as the minimization of a convex objective function under special structure constraints (order constraints); it is solved by a new efficient algorithm that considerably reduces

the computational effort, in the presence of a large amount of data, needed to evaluate the objective function.

We then present deterministic and robust models around the optimal management of water networks using an automated strategy based on the use of trigger levels. We also show that the problem can be reduced to smaller daily problems without losing too much accuracy.

Numerical results are presented and reveal the possibility of strong savings in the different energy systems studied.

**Keywords :** *Data science, Energy optimization, Mathematical programming, Robust optimization, Uncertainties.*