Université de Liège

Gestion de délestage d'électricité

Mémoire de fin d'études réalisé en vue de l'obtention
du grade d'Ingénieur Civil Électricien

Promoteurs : Q. Louveaux, D. Ernst

Année académique 2011-2012

MATHIEU Sébastien

Je remercie tout particulièrement la société *POWERDALE*, en particulier *Ludovic Amand* et *Olivier Piraux*, pour m'avoir donné l'opportunité de travailler avec leur collaboration sur ce problème très intéressant. Plus que d'apporter le sujet, ils ont amené une dimension pratique à ce travail.

Je tiens également à adresser mes remerciements à mes promoteurs pour leur disponibilité et l'intérêt qu'ils ont portés à ce travail de fin d'études tout au long de l'année.

Enfin, je remercie ma correctrice chevronnée pour la révision de ce texte.

Table des matières

1 Introduction

"Sauvez la planète !", *"Les éoliennes ? C'est pas sorcier !"*, *"Il n'y a pas de planète B"* : ces slogans nous apparaissent au quotidien, ils sont synonymes de respect de l'environnement, d'économies, de responsabilité ... L'idée est très claire dans l'esprit de tout habitant de la terre ; on en parle aussi bien au journal qu'à l'école primaire : il faut réduire notre *empreinte écologique* sur la planète. Une partie de la solution consiste à modifier la manière dont l'électricité est produite.

Le nucléaire dérange de plus en plus au fil des catastrophes telles que *Chernobyl* et *Fukushima*. Le pétrole et le gaz commencent à se raréfier et la dépendance énergétique est cause de crises géopolitiques de très grande ampleur. L'alternative proposée par l'énergie renouvelable s'impose donc d'elle-même. Cependant, le réseau électrique a été conçu à une époque où ces solutions n'existaient pas encore. L'*économie d'échelle* régnait en maître et on n'aurait pas imaginé un jour remplacer les centrales nucléaires par un réseau composé de petites unités délocalisées produisant une énergie aussi imprévisible.

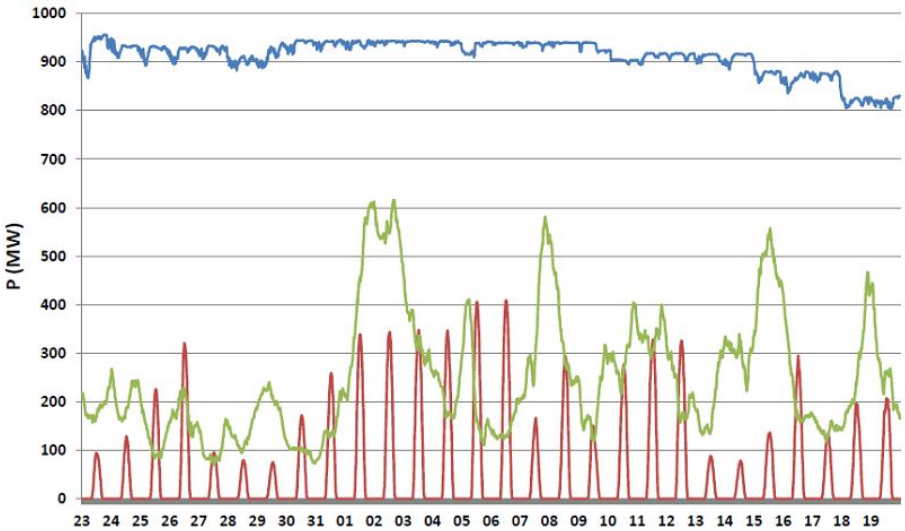

FIGURE 1.1 – Période du 23/01 au 19/02 2012. Toutes les courbes indiquent, pour un moyen de production donné, la puissance qu'il a livrée au réseau (en MW) par GW de puissance installée (1GW=1000MW) en France ou en Allemagne. La courbe bleue correspond au Nucléaire France, la courbe verte à l'éolien France et la courbe rouge au solaire PV Allemagne. [5]

En effet, malgré les progrès de plus en plus impressionnants en matière de météorologie, on ne sait toujours pas prévoir exactement l'ensoleillement pour le lendemain et ainsi connaitre parfaitement

la production par énergie solaire. La production d'énergie éolienne est, quant à elle, aussi erratique qu'imprévisible. Au final, le réseau électrique se retrouve tantôt saturé, tantôt en sous-production, et malheureusement, la transition entre ces deux états peut ne durer qu'un quart d'heure. Si l'on peut espérer raisonnablement une amélioration de la prévision, cela ne résoudra pas le problème du comportement irrégulier de la production.

Pour l'instant, on choisit d'adapter la production à l'aide d'unités de réglage, c'est à dire utiliser des machines capables de faire varier la puissance qu'elles fournissent très rapidement. Actuellement, ces unités de réglage sont généralement alimentées par des énergies fossiles, ce qui assombrit donc le tableau d'un parc électrique totalement *vert*. Évidemment, le prix au mégawatt de ce genre d'unités est beaucoup plus élevé que celui de grosses unités de production telles que les centrales nucléaires. Au-delà de ce coût direct, il faut considérer celui dû à la réservation de ces unités pour un service auxiliaire au lieu d'être utilisées directement en production : cela entraîne donc un sur-dimensionnement de la production par rapport aux capacités réelles disponibles. En effet, s'il n'y a aucune fluctuation dans le réseau, ces unités ne seront pas utilisées du tout car devant être réservées pour leur tâche de régulation.[9] A l'heure actuelle, ces coûts restent raisonnables grâce à une grande stabilité de la production, mais avec l'introduction des énergies vertes, ce ne sera plus le cas.

Une alternative vient tout naturellement : **si la production ne peut s'adapter, c'est à la charge à évoluer**. *"Une gestion intelligente du réseau électrique alimentée par les données de consommation et production en temps réel permettra donc un meilleur ajustement de la production et de la consommation d'électricité"* [2]. Le stockage par des unités centralisées s'avère être peu efficace, à l'exception de centrales de pompage-turbinage telle que *Coo*. Malheureusement, des centrales telle qu'à *Coo* nécessitent une topographie particulière et les sites répondant aux critères indispensables pour obtenir un réel gain ne sont pas légion. Dans le futur, ce genre de système pourrait même devenir inenvisageable de par la taille d'une telle structure combinée à la hausse du prix du terrain.

Heureusement, une solution décentralisée est envisageable. Utilisons certaines charges du réseau électrique pour contribuer à ce dernier. L'idée de mettre à contribution les charges du réseau électrique n'est pas nouvelle. Depuis bien des années, on parle de délestage des charges comme en atteste l'article [11] datant de 1971. Effectivement, beaucoup de systèmes électriques sont des unités de stockage prêtes à être exploitées, tels que les chauffages ou encore les voitures électriques. Une pièce stocke naturellement la chaleur et tant que sa température reste acceptable, on peut choisir de chauffer quand bon nous semble. Dans le cas des voitures électriques, les batteries de celles-ci sont des unités de stockage évidentes d'énergie électrique. A l'avenir, le parc de voitures électriques va sans aucun doute se développer davantage, offrant ainsi une capacité de stockage formidable [14]. La vraie question maintenant est "Comment exploiter cette capacité des charges encore latente jusqu'à maintenant ?".

Avant d'y répondre, un bref rappel sur l'organisation du réseau électrique belge peut être utile. Les lignes sont administrées par le **gestionnaire du réseau électrique**. Le consommateur achète son électricité à une **compagnie de distribution**. Les différentes compagnies de distributions achètent, pour le lendemain, leur électricité via un marché de gros aux **compagnies de génération**. Finalement, le gestionnaire du réseau électrique s'assure que l'équilibre entre l'offre et la demande est atteint. Si ce n'est pas le cas, celui-ci a accès à des services auxiliaires achetés par le gestionnaire du réseau sur un marché spécifique. En effet, afin d'obtenir un marché privatisé, le gestionnaire du réseau ne peut pas légalement posséder d'unités de production qui, comme mentionné, servent à équilibrer la production et la consommation.[6]

Mais le marché des services auxiliaires est également accessible à des sociétés autres que les fournisseurs. Ainsi commencent à se créer des entreprises *d'agrégation de charges*. Celles-ci achètent la flexibilité de charges de manière à se constituer un parc sur lequel elles ont une certaine emprise, c'est à dire la possibilité de les manœuvrer. C'est en dirigeant cet ensemble de charges que, une fois celles-ci agrégées, ce type d'entreprises peut vendre un service auxiliaire. Notons que la tâche d'accès aux charges n'est pas dénuée d'embuches : cela pose de gros problèmes de sécurité et de confidentialité [12] [13].

Une des sociétés émergentes sur ce marché porteur est Powerdale qui est à l'origine de ce travail. Elle a notamment accès à plusieurs parcs de voitures électriques et espère utiliser celles-ci dans le domaine des *réseaux électriques intelligents*. Celle-ci a pour but de vendre la flexibilité de ces charges au travers d'un services de modulation de charges vendu, par exemple, au gestionnaire du réseau de transport. Dans ce travail, nous considérons un service spécifique qui consiste en une option vendue un jour à l'avance par un aggrégateur. Cette option détermine une quantité de réglage, c'est à dire une modulation de la puissance consommée par les charges gérées par l'aggrégateur, à la hausse ou à la baisse, ainsi que la durée d'une action de réglage. Une fois l'option en sa possession, le client peut demander à tout moment de la journée, une modulation de la charge de l'aggrégateur.

Mais qu'entend-on précisément par *"réglage"* ? On peut définir cette action comme une série de décisions menant à un changement de la quantité de puissance consommée par l'ensemble des charges. Si cette puissance totale augmente, on aura un réglage à la hausse. A l'inverse, si elle diminue, on aura un réglage à la baisse ou délestage. La différence de puissance totale sera appelée une *quantité de réglage*.

La première application de ces délestages est, comme décrit dans [19], la déconnexion de charges du réseau électrique dans le cas où la production d'électricité est insuffisante. Si l'on conservait toutes les charges sur le réseau, la fréquence du réseau chuterait et cette perte de synchronisme ferait décrocher une à une toutes les unités de production jusqu'à plonger des pays entiers dans le noir. Évidemment, cette mesure n'est utilisée qu'en dernier recours, en commençant par les habitations et en terminant par les industries. Ce type de délestage a déjà été le fruit de nombreuses recherches, à commencer par la détermination de l'amplitude du délestage nécessaire afin de rétablir la fréquence du réseau, comme étudié dans [18]. Si l'on peut connaitre la quantité à délester, un système complémentaire est nécessaire pour effectivement fournir les charges à délester. Dans ce but, des études ont été effectuées afin d'utiliser certains types de charges spécifiquement, tels que l'air conditionné et les réfrigérateurs [17] ou les lumières [15]. Grâce à la constante décroissance du prix de l'électronique, on entend de plus en plus parler de la notion de *"smart meter"*, ces petits boitiers destinés, dans un premier temps, à l'observation de la consommation électrique, mais ultimement à la commande des appareils connectés. Encore une fois, le délestage en cas de sous-tension a déjà été étudié, comme en atteste entre autres [16]. En bref, comme le confirme [3], la réaction de la demande peut faire office de réserve sur le réseau électrique.

Non plus dans le domaine de la réserve mais dans le fonctionnement normal du réseau électrique, le *peak shaving* permet de réduire la puissance maximum consommée sur le réseau afin de réduire les coûts de dimensionnements. En effet, dans les réseaux électriques, on prévoit suffisamment d'unités de production pour être capable de satisfaire la plus grande consommation d'électricité. On a donc tout intérêt, si l'on veut réduire l'équipement nécessaire, à ne pas obtenir de pics de consommation. Par conséquent, des mesures de délestages sont aussi les bienvenues dans le fonctionnement normal du réseau. Citons notamment une étude générale basée sur la possibilité d'actions à partir de charges dites *interruptibles* [4].

Si l'intérêt du délestage est évident pour le gestionnaire du réseau, il ne faut pas oublier que les entreprises elles-mêmes ont intérêt à contrôler judicieusement leur consommation de manière à obtenir une consommation électrique la plus constante possible. Un exemple pratique est donné pour une exploitation minière dans l'article [10]. Celui-ci contient une formulation sous la forme d'un problème d'optimisation linéaire en nombres entiers permettant de déterminer les actions à entreprendre afin de réaliser un délestage et la restauration des charges, si cela est nécessaire.

Deux missions

Dès lors, il est aisé de comprendre le but de ce travail : **connaître et maximiser les quantités de réglages à la hausse et à la baisse que l'aggrégateur est capable d'offrir**, et ce, en jouant sur la flexibilité des différentes charges du portefeuille de l'aggrégateur. Si l'on prend le cas simple d'une

voiture électrique restant sur son socle de charge 10 heures, alors que le temps réellement nécessaire pour obtenir une batterie pleine n'est que de 5 heures. Sur cette période de 10 heures, nous **choisirons** alors les moments pendant lesquels la batterie sera alimentée tout en assurant sa charge complète.

Ce sont donc ces *choix* qu'il va falloir effectuer le plus intelligemment possible avec pour objectif de maximiser les quantités de réglage, à la hausse et à la baisse, disponibles tout au long d'une journée. Nous subdiviserons une journée en plusieurs périodes, par exemple, 96 périodes de 15 minutes. Une demande de réglage pourra survenir à n'importe quelles de ces périodes. Celle-ci sera sollicitée par le gestionnaire de réseau ayant acheté l'option à *aggrégateur* qui se chargera d'envoyer les commandes nécessaires à son parc de charges afin de satisfaire cette demande.

Notre mission est donc, premièrement, d'effectuer une prévision de la quantité de réglage à la hausse et à la baisse que l'on peut assurer à chaque période de temps. Grâce à cette information, l'*aggrégateur* connaîtra ce qu'elle peut effectivement vendre en exploitant son parc de charges. Ensuite, il sera nécessaire d'être capable d'effectuer les décisions sur les différentes charges afin d'atteindre une quantité de réglage donnée. Ce travail se préoccupera donc non seulement de l'action, mais également de la prévision de celle-ci.

Modélisation

La première difficulté réside dans le fait que nous devons gérer différents types de charges qui ne se comportent pas toutes de la même manière. Afin de rester flexible, il est nécessaire d'établir une couche d'abstraction identifiant toutes les charges à un même patron. Voitures électriques, pompes à chaleur, systèmes de remplissage de cuves, ... toutes ces charges seront assimilées à un **système de stockage générique**. Ce système de stockage permet de considérer toute charge comme un réservoir dont le remplissage est une fonction de la puissance électrique donnée en entrée.

Une fois les charges modélisées, il est alors nécessaire d'aboutir à un modèle d'optimisation mathématique établissant le lien entre les puissances associées à celles-ci et les quantités de réglage qui pourront être vendue sous la forme d'un service auxiliaire, le produit final. Le modèle doit permettre à l'aggrégateur de connaître comment gérer son portefeuille de charges et savoir les quantités de réglages exactes qu'il sera capable de fournir tout au long de la journée à venir. Finalement, on aboutit à un problème qui pourra être résolu, entre autres, à l'aide de la programmation linéaire en nombres entiers.

Prévision des quantités de réglage

Obtenir une formulation du problème est inutile si on est incapable de le résoudre. Il existe un algorithme fort efficace pour ce genre de problèmes pouvant nous donner une solution qui a l'énorme avantage d'être optimale si on lui laisse assez de temps. Mais le problème est là, aucune garantie ne nous est donnée au niveau temps. La taille du problème devient vite beaucoup trop importante que pour se cantonner à cette seule solution.

De plus, il est préférable de ne pas dépendre d'outils tiers et d'obtenir une méthode relativement simple de manière à pouvoir en assurer la maintenance, si possible développée en *Java*. Trois idées ont donc été explorées pour tenter d'approcher au mieux la solution optimale en un minimum de temps. Quelques tests permettent de juger leurs résultats et de sélectionner le meilleur d'entre eux.

La première idée pour résoudre le problème est l'agrégation de toutes les charges en un seul réservoir d'une capacité équivalente, de déterminer un *niveau cible* à atteindre pour chaque période de temps. Une fois celui-ci fixé, on tente de répartir ce niveau global sur l'ensemble des charges du parc. A ce moment, il est indispensable d'assurer que la solution construite est bel et bien **réalisable** pour chaque charge, c'est à dire qu'elle doit respecter l'ensemble des contraintes, par exemple, ne pas obtenir un réservoir qui déborde.

Au lieu de se baser sur un niveau cible, la seconde idée s'appuie sur une *puissance totale cible* qui va encore une fois être redistribuée entre les différentes charges, et ce, de manière à obtenir une solution réalisable. Si obtenir une solution réalisable n'est plus un problème, celui-ci ayant été résolu à l'élaboration du premier algorithme, la difficulté se trouve ici dans la recherche d'une méthode efficace afin de déterminer quelle charge allumer et quelle charge éteindre.

Finalement, la troisième idée suivie a été de ne plus prendre une décision globale sur l'ensemble des charges, mais de chercher des *faiblesses* dans une solution pré-existante. Qu'entend-on par faiblesse ? Il s'agit d'une charge qui, à un moment où elle devrait effectuer un réglage à la baisse, effectue un réglage à la hausse alors que la quantité de réglage totale à la baisse est peu importante. Lorsque cette faiblesse est identifiée, il faut alors corriger la solution intelligemment.

Une fois ces algorithmes développés, nous prenons également en compte l'**incertitude** inhérente au problème. En effet, si l'on considère l'exemple d'une voiture électrique, on ne peut savoir si celle-ci utilisera, pour son trajet, 50% ou 60% de sa batterie. De manière générale, les données ne peuvent pas toujours être connues avec précision. Les méthodes élaborées ont donc été étendues à la prise en compte de cette incertitude.

Décisions pour obtenir une quantité de réglage précise

A l'aide des algorithmes développés précédemment, nous connaissons les quantités maximales de réglage que nous sommes capables de fournir. Maintenant, il faut répondre au second problème : *connaissant la période à laquelle commencer le réglage et la quantité à atteindre, comment activer les différentes charges du parc ?*

Il est donc question de la fourniture effective du service de modulation de la charge et non plus de s'assurer sa disponibilité. Si, précédemment, on recherchait la quantité de réglage maximum, on peut désirer atteindre une valeur inférieure pour, par exemple, obtenir un équilibre global sur le réseau entre production et consommation d'électricité. Dans le cas d'un délestage maximal, on tentait de couper l'alimentation de l'ensemble des charges. Mais si maintenant, on désire diminuer la puissance dans une bien plus petite proportion, il va falloir déterminer quelles charges activer et quelles charges couper de manière à obtenir la quantité de réglage désirée.

La structure de ce document est la suivante : dans le chapitre deux, nous nous concentrerons sur la modélisation des charges du portefeuille de l'aggrégateur. Nous enchainerons avec la modélisation complète du problème. Dans le quatrième chapitre, nous établirons une méthode permettant de connaitre si le problème est réalisable. Les cinq chapitres suivants traiteront de méthodes de résolutions simples et s'achèveront avec quelques tests dédiés à apprécier la performance de celles-ci. Nous traiterons alors une méthode pour obtenir une quantité de réglage précise. Finalement, nous verrons comment ajouter l'incertitude inhérente aux données à ce qui aura été développé dans ce travail.

2 Modélisation des charges

2.1 Modélisation

Beaucoup de charges fort différentes peuvent se trouver dans le portefeuille d'un aggrégateur : des voitures électriques, des pompes à chaleurs, ... L'objectif de cette partie est de permettre l'exploitation de la flexibilité des charges du portefeuille de l'aggrégateur au sein d'un système d'optimisation et donc d'établir un modèle mathématique représentant chaque charge. L'accès à la charge se fait par l'intermédiaire d'un *logger* permettant de rapatrier les différentes données caractérisant la charge. Ce *logger* permet éventuellement le réglage de l'alimentation de cette dernière. C'est à partir de ce point d'accès que le système développé ici va interagir avec son environnement physique. On pourrait imaginer une communication basée par exemple sur [8].

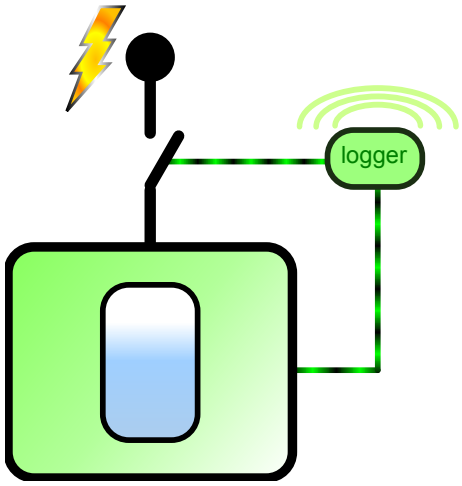

FIGURE 2.1 – Accès à la la charge via le *logger*.

La modélisation que nous proposons établi un modèle à la fois général pour modéliser fidèlement un grand nombre de charge et simple pour être aisément compréhensible et applicable. Nous nous référons au trait commun à toutes ces charges : elles sont toutes assimilables à un réservoir dont le niveau doit être compris entre une borne minimale et une maximale. Ce système de réservoir générique est schématisé à la figure 2.2.

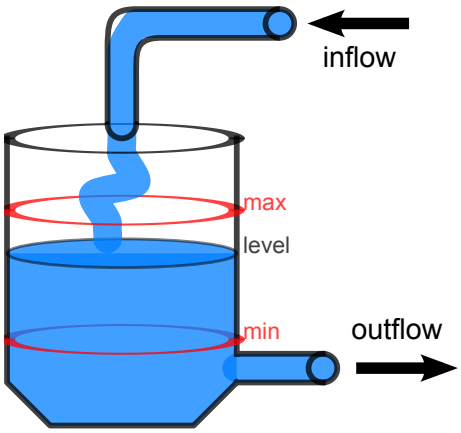

FIGURE 2.2 – Système de stockage d'énergie générique.

Ce système de stockage générique (d'où l'omission des unités) est **entièrement** déterminé par les paramètres suivants pour une charge i au temps t :

- Niveau minimal $min_{i,t}$ et maximal $l_{i,t}^{max}$.
- Flux de sortie $out_{i,t}$.
- Consommation d'énergie électrique minimale si la charge est activée $l_{i,t}$.
- Consommation d'énergie électrique maximale si la charge est activée $u_{i,t}$.
- Décalage temporel δ_i entre l'activation de la charge et son effet sur le niveau.
- Deux coefficients établissant le lien entre le flux d'entrée et la consommation d'énergie électrique : $a_{i,t}$ et $b_{i,t}$.

L'utilisation de deux coefficients traduit la linéarisation de la loi donnant le flux d'entrée en fonction de la consommation électrique qui n'est pas nécessairement linéaire. Cependant, nous supposerons qu'il s'agira d'une bonne approximation dans notre plage de fonctionnement (en principe réduite).Le niveau d'une charge i au temps $t+1$, noté $level_{i,t+1}$, est donnée par :

$$level_{i,t+1} = level_{i,t} - out_{i,t} + (a_{i,t-\delta}.power_{i,t-\delta} + b_{i,t-\delta}.on_{i,t-\delta})$$

Par facilité, on considèrera également que $a_{i,t} \geq 0$ avec $a_{i,t} = 0 \Rightarrow b_{int} = 0$ pour tout t.

Mentionnons également la possibilité d'établir une priorité pour certaines charges (via un coefficient c dans la modélisation). Plus ce coefficient sera positif, plus la charge sera préférentiellement activée. A contrario, une charge ayant un coefficient négatif sera de préférence désactivée.

2.2 Applications

Cette partie permet de donner des exemples de modélisations de charges *pseudo-réelles* comme les réservoirs génériques présentés dans le modèle. Ceci permet de montrer que la démarche est réalisable. Nous tenterons ici de générer des données vraisemblables pour des voitures électriques et des chauffages de manière.

2.2.1 Voitures électriques

Idée générale

Nous distinguerons deux cas :

- La voiture est présente sur son socle de charge.

– La voiture est en déplacement.

Dans le premier cas, le flux d'entrée suivra une loi classique avec les coefficients a *et* b, sinon ces deux coefficients seront nuls. Le niveau de la batterie sera donné en pourcents avec un niveau minimal à déterminer (par exemple 10%). L'heure de départ de la voiture ainsi que la distance à parcourir étant toutes deux supposées programmées, on peut définir, pour la période précédent cette heure critique, une valeur minimale du niveau de la batterie à atteindre afin de garantir une charge nécessaire pour le trajet prévu. Lors de son absence du socle, nous n'avons aucune prise sur la voiture, et ce n'est qu'à son retour que l'on déduira la consommation de la voiture via la variable out.

Nous faisons l'approximation que si l'on charge la batterie à une puissance donnée, la variation de niveau qui suivra sera identique quel que soit la charge initiale de la batterie. En pratique, il est connu que ce n'est pas le cas, mais une valeur moyenne peut en première approximation être utilisée. De plus, une modélisation plus fidèle pourra être obtenue via l'utilisation de l'incertitude qui sera abordée dans le chapitre 10.

Calcul des coefficients a en charge

Admettons comme connu le niveau minimal de la batterie (noté l_0), le temps de charge pour passer du niveau minimal à pleine charge (noté θ), la puissance moyenne en entrée (notée P_{in}) et les pertes en terme de niveau (notée p), ces deux dernières grandeurs étant calculées sur la période de temps considérée (par exemple 15 minutes).

Le niveau en tout temps t où la voiture est présente est donné par :

$$l_t = l_{t-1} + aP_{in} - p$$

où l'on comprend que le coefficient b est en quelque sorte inclus dans cette valeur mesurée p (autrement écrit : $b = -p$).

A partir du niveau initial l_0, on a, au bout du temps de charge :

$$l_\theta = l_0 + \sum_{t=0}^{\theta-1}(aP_{in} - p) = l_0 + \theta(aP_{in} - p)$$

Et en isolant a, nous trouvons :

$$a = \frac{\frac{l_\theta - l_0}{\theta} + p}{P_{in}}$$

Implémentation : *LoadElectricCar*

Une fois que l'on a les paramètres permettant d'associer une voiture électrique à un *système de stockage d'énergie générique*, c'est à dire que l'on a obtenu les coefficients a et b de la régression linéaire, sa modélisation est aisée.

On définit une probabilité d'utilisation de la voiture électrique P_{usage} et à toute période, on va générer un nombre aléatoire entre 0 et 1. Si ce nombre est supérieur à la probabilité d'utilisation, la voiture reste sur son socle de recharge. Sinon, celle-ci va être utilisée pour un trajet dont la distance sera tirée au sort.

Afin de définir une utilisation, il faut prévoir la durée pendant laquelle la voiture sera absente et la charge minimale nécessaire au trajet. Cependant, cette dernière donnée peut ne pas être possible à déterminer si la voiture n'a pas eu le temps nécessaire pour recharger depuis sa dernière utilisation. On veillera donc à définir une plage de possibilités pour le niveau minimal à atteindre.

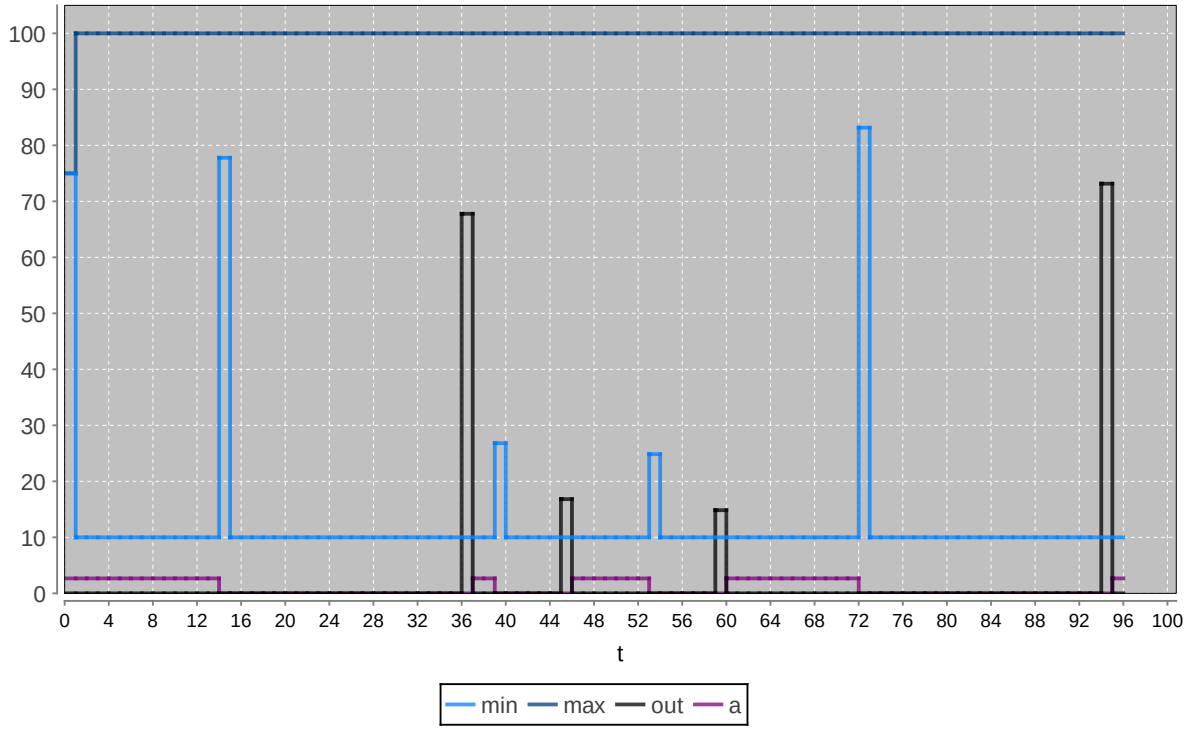

FIGURE 2.3 – Exemple de données générées d'une voiture électrique en fonction du temps (96 périodes d'un quart d'heure).

Le niveau minimal à atteindre le plus élevé est logiquement celui où, depuis la dernière remise à son socle, on aurait chargé la voiture à pleine puissance à partir du niveau de la batterie à cet instant.

Un exemple de données générées est donné à la figure 2.3. On y voit un niveau minimal constant (en bleu clair) à l'exception des différents départs de la voiture identifiables par les pics. On repère sans difficultés les pics de consommation (en noir) correspondant aux retours de la voiture. La voiture est sur son socle lorsque le facteur a (en mauve) est différent de 0.

2.2.2 Chauffage

Idée générale

Nous allons considérer une pièce comme un réservoir dont le niveau sera mesuré par sa température. On désirera avoir, durant les heures de travail, une température comprise dans une certaine plage. En dehors de ces heures, une plage de température beaucoup plus large sera définie.

Cette fois, nous définirons un intervalle de puissance dans lequel le chauffage pourra fonctionner. La perte de température en fonction du temps, tout comme l'effet du chauffage en terme de degrés, doivent être évalués au cas par cas.

Implémentation *LoadHeater*

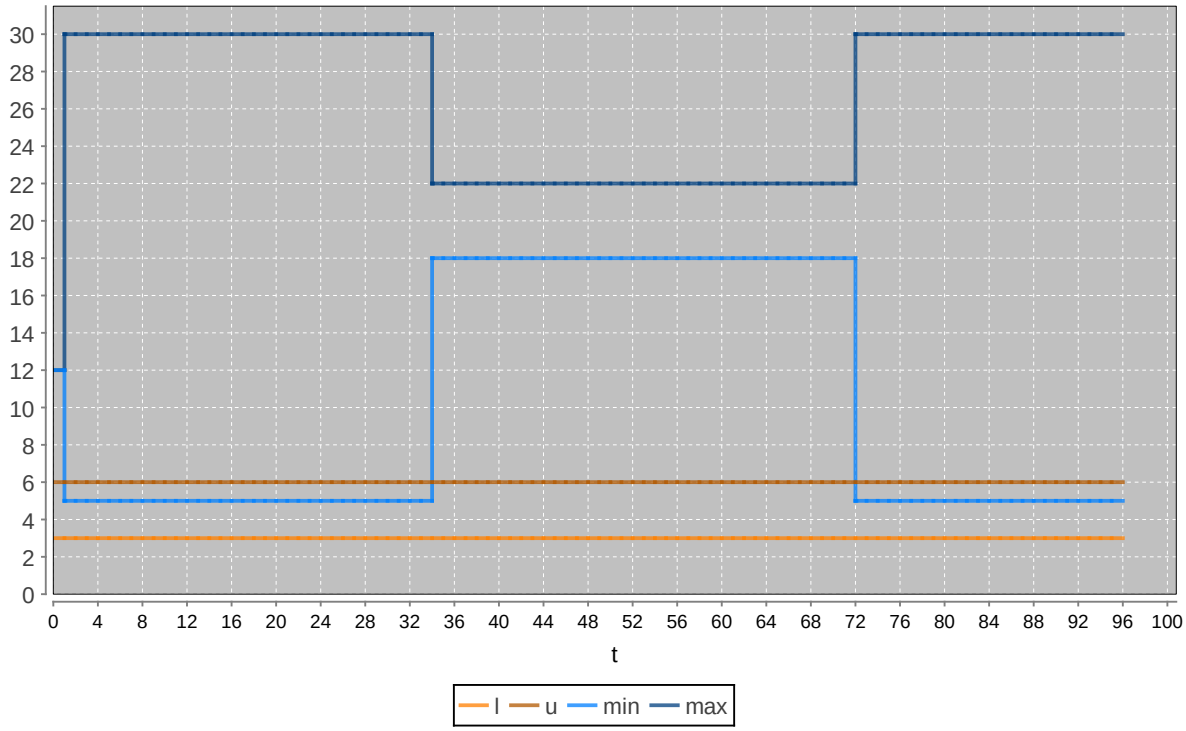

FIGURE 2.4 – Exemple de données générées d'un chauffage en fonction du temps (96 périodes d'un quart d'heure).

On observe, à la figure 2.4, un graphique représentant un chauffage avec en bleu clair la température minimale, en bleu foncé la température maximale. La puissance maximale est tracée en brun et la minimale en orange.

3 Modélisation du problème

3.1 Énoncé général du problème

Dans cette partie, nous modélisons le problème décisionnel rencontré par un aggrégateur désireux de connaitre et maximiser les quantités de réglage à la hausse et à la baisse qu'il est capable de fournir à l'aide la gestion de son portefeuille de charge. Le produit que l'aggrégateur va vendre consiste en un service de modulation de charge. Ce service est vendu sous la forme d'une option un jour à l'avance au gestionnaire du réseau de transport. L'option spécifie la quantité maximale de réglage à la hausse et à la baisse qui peut être demandée ainsi que la durée d'une modulation. L'option peut être utilisée par le gestionnaire du réseau de transport à tout moment de la journée, mais seulement une seule fois par jour.

Prenons pour exemple la figure 3.1 qui représente schématique un cas où le gestionnaire du réseau de transport spécifie à l'aggrégateur, dans le quart d'heure allant de $14:30$ à $14:45$, de délivrer un réglage à la baisse de $80\ MW$ pour 30 minutes. En se basant sur la consommation totale de son parc de charges de $200\ MW$ au moment de la demande de réglage, l'aggrégateur doit assurer que la puissance totale consommée par son portefeuille de charges entre $14:45$ and $15:15$ est égale à $120\ MW$.

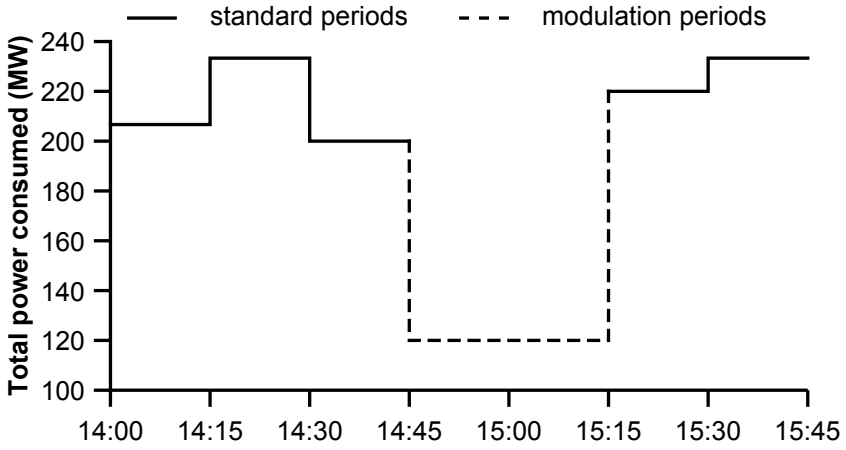

FIGURE 3.1 – Exemple où le TSO utilise son option de réglage pour un réglage à la baisse de $80\ MW$ pendant une demi heure à partir de $14:45$.

A partir des données acquises concernant les différentes charges, le système devra donc être capable de déterminer quelles sont les décisions optimales à prendre afin de maximiser les quantités

de réglages que l'aggrégateur sera capable de fournir et ce dans toutes les situations, c'est à dire pour tout instant ou le gestionnaire du réseau de transport peut demander une modulation. Il parviendra ainsi à déterminer les charges à activer au moment propice et l'on connaitra les capacités totales de réglage à la hausse et à la baisse résultantes (en terme de puissance).

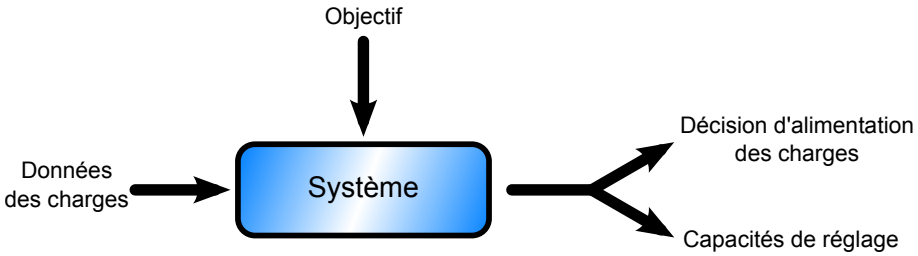

FIGURE 3.2 – Aperçu du rôle du système

Afin de s'approcher au mieux de la solution optimale, il a été choisi de partir sur une modélisation mathématique de manière à être résolu par la théorie de l'optimisation discrète. Cependant, les librairies nécessaires à la résolution de tels problèmes sont en général couteuses pour obtenir les performances souhaitées. C'est pourquoi, dans la seconde partie du travail, est exposée une méthode de résolution s'affranchissant de ce genre de librairies annexes.

3.2 D'un problème réel à un problème mathématique

3.2.1 Demandes de réglage

On entend par *"Demandes de réglage"*, une requête dont l'objectif sera de faire varier la puissance consommée globale en diminuant ou en augmentant la charge du réseau électrique. Une demande de réglage à la baisse, ou **délestage**, se traduira par la réduction, voire même la coupure, de la consommation de certaines charges. On peut citer, par exemple, l'extinction d'un chauffage pour une demi-heure. A l'inverse, une demande de réglage à la hausse se traduira par l'activation ou l'augmentation de certaines charges, comme activer la charge de voitures électriques.

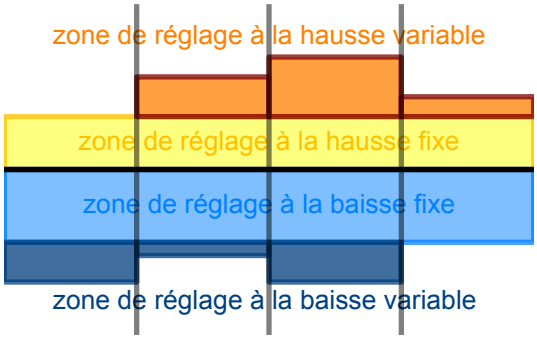

FIGURE 3.3 – Illustrations des différentes zones de réglages potentielles pour quatre périodes de temps.

Définissons mathématiquement la notion de réglage. Considérons la puissance consommée en deux temps : t et $t+1$, la différence entre les deux $\Delta = P_t - P_{t+1}$ correspondra à un réglage à la hausse si cette valeur est positive, sinon à un réglage à la baisse. Les demandes de réglage peuvent se diviser en quatre zones : deux zones de réglage fixes et deux zones de réglages variables. Pour chaque type

de réglage, on définira la zone de réglage fixe comme étant les minimums de quantités de réglage possibles sur toutes les périodes de temps . Cette grandeur nous intéresse particulièrement car on peut, à partir de sa connaissance, vendre une *option* de réglage disponible tout au long d'une journée par exemple.

Finalement, ces demandes de réglages dont l'importance est mesurée par la quantité de réglage associée est l'objet de ce travail de fin d'étude.

3.2.2 Scénarios

La spécificité du service étudié repose sur le fait que l'on ne sais pas préalablement quand le gestionnaire du réseau électrique va effectuer une demande de réglage. Nous devons donc considérer tous les cas, c'est à dire être capable de fournir à toute période où une demande de réglage est possible, une même quantité de réglage. Nous résolvons donc le problème du délestage sur un certain nombre de scénarios, un par demande de réglage possible sachant que le gestionnaire du réseau ne peut effectuer une demande qu'une seule fois par jour.

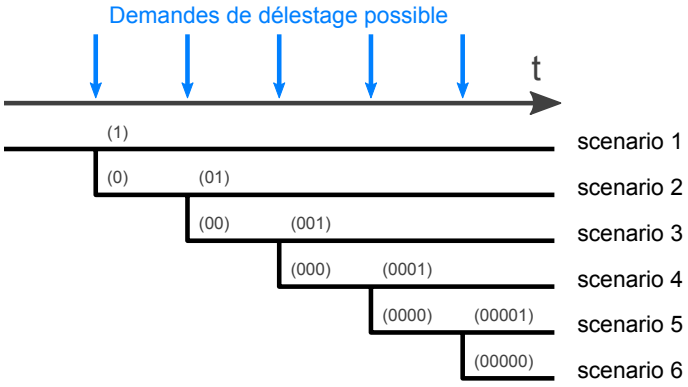

FIGURE 3.4 – Scénarios ordonnés en fonction du moment du délestage

On ordonne les scénarios de telle manière qu'un scénario i soit "inférieur" à un scénario j si le délestage de i arrive en un temps T_i et le délestage de j arrive en T_j tel que $T_i < T_j$. Une simplification apparait alors : pour les temps $t < T_i$, l'état du système est le même pour les deux scénarios. Dans l'explication précédente, on a considéré uniquement les réglages à la baisse. Pour prendre en compte les réglages à la hausse, il va falloir doubler le nombre de scénarios. En effet, on va considérer que l'on peut également faire un seul réglage à la hausse sur une journée.

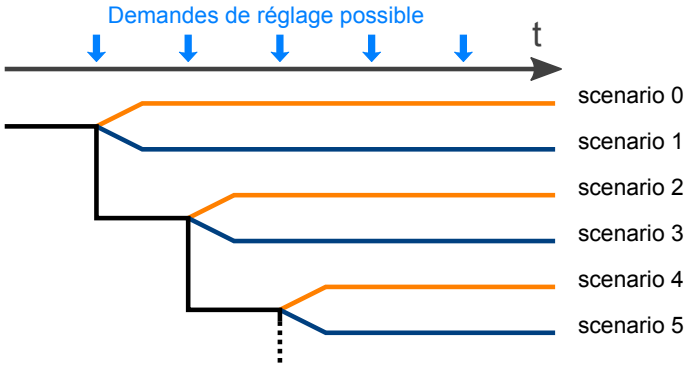

FIGURE 3.5 – Scénarios ordonnés en fonction du moment du réglage

On va envisager le pire des cas, c'est à dire que l'on considèrera un réglage à la hausse indépendant de tout réglage à la baisse potentiellement produit dans une période antérieure. En effet, si l'on avait effectué un réglage à la baisse à la période T, il serait d'autant plus facile de faire un réglage à la hausse en $T + i$ pour $i \in \mathbb{Z}_0^+$. Évidemment, la logique est identique pour le cas d'un réglage à la hausse se produisant avant un réglage à la baisse.

Afin de respecter le système de scénarios "inférieurs", on définira les indices j impairs pour des réglages à la baisse et pairs pour des réglages à la hausse.

3.3 Modèle

3.3.1 Données

$min_{i,t}$	Niveau minimal de la charge i
$max_{i,t}$	Niveau maximal de la charge i
$out_{i,t}$	Flux de sortie de la charge i au temps t
$a_{i,t}; b_{i,t}$	Coefficients tels que le flux d'entrée pour la charge i au temps $t + \delta$ soit égal à $a_{i,t} \times power_{i,t} + b_{i,t}$
$l_{i,t}$	Puissance minimale demandée par la charge i au temps t
$u_{i,t}$	Puissance maximale demandée par la charge i au temps t
δ_i	Délai sur le flux d'entrée par rapport au niveau pour la charge i
$c_{i,t}$	Coefficient de priorité de la charge i au temps t
N	Nombre de périodes de délestages consécutives demandées
S	Nombre de scénarios

On notera l'ensemble des nombres allant de 0 à S pairs, donc correspondant aux réglages à la **h**ausse, comme l'ensemble S_h, et les impairs S_b. Par facilité, on considèrera également que $a \geq 0$ avec $a = 0 \Rightarrow b = 0$.

3.3.2 Variables

$on_{i,t}^{(j)}$	Binaire égale à 1 si la charge i est active au temps t dans le scénario j
$level_{i,t}^{(j)}$	Niveau du réservoir de la charge i au temps t pour le scénario j
$power_{i,t}^{(j)}$	Puissance électrique demandée par la charge i au temps t dans le scénario j
Q_{fi}^{+}	Capacité de réglage à la hausse minimale
Q_{fi}^{-}	Capacité de réglage à la baisse minimale
$Q_{va_t}^{(j)}$	Capacité variable pour un réglage demandé en T_j pour le scénario j
$Q_{va_t}^{+}$	Capacité variable minimale pour un réglage à la hausse
$Q_{va_t}^{-}$	Capacité variable minimale pour un réglage à la hausse

3.3.3 Objectif

Profit

Maximiser le profit de manière robuste tout en assurant un certain confort au client et en laissant la priorité quant à l'activation de certaines charges.

$$\max R_{pr} + R_{act} + C_{loads}$$

Avec,

$$C_{loads} = \sum_{i}^{} \sum_{j}^{S} \sum_{t}^{} c_{i,t} \text{on}_{i,t}^{(j)} \qquad \text{Coût fictif de la charge}$$

$$R_{pr} = R_{pr_{fi}} + R_{pr_{va}} \qquad \text{Recettes de mise à disposition}$$

$$R_{pr_{fi}} = Q_{fi}^{+} T_{fi}^{+} + Q_{fi}^{-} T_{fi}^{-} \qquad \text{Zone de réglage fixe}$$

$$R_{pr_{va}} = \sum_{t}^{T-1} Q_{va_t}^{+} T_{va_t}^{+} + \sum_{t}^{T-1} Q_{va_t}^{-} T_{va_t}^{-} \qquad \text{Zones de réglage variable}$$

$$R_{act} = \sum_{t}^{T-1} \sum_{j \in S_h}^{} P_{act_j} \left(Q_{fi}^{+} + Q_{va_t}^{+} \right) \qquad \text{Recettes pour l'activation}$$

Tel que :

T_{fi}^{+}	Prix de mise à disposition de réglage à la hausse de la zone fixe
T_{fi}^{-}	Prix de mise à disposition de réglage à la baisse de la zone fixe
$T_{va_t}^{+}$	Prix de mise à disposition de réglage à la hausse de la zone variable au temps t
$T_{va_t}^{-}$	Prix de mise à disposition de réglage à la baisse de la zone variable au temps t
$T_{act_t}^{+}$	Prédiction du prix marginal d'activation de réglage à la hausse au temps t
$T_{act_t}^{-}$	Prédiction du prix marginal d'activation de réglage à la baisse au temps t
P_{act_j}	Probabilité que le scénario j se produise

Quantité de réglage

Très similaire au précédent avec un coefficient C a dimensionner tel que $C \gg c_{i,j}\ \forall i,j$ pour que les quantités de réglages soient prioritaires par rapport au bien-être des charges :

$$\max C \left[T \left(Q_{fi}^{+} + Q_{fi}^{-} \right) + \sum_{t=0}^{T-1} \left(Q_{va_t}^{+} + Q_{va_t}^{-} \right) \right] + C_{loads}$$

Dimensionnement du coefficient C

L'idée principale est de dimensionner le coefficient C à partir des coefficients $c_{i,j}$ tel que :

$$CQ_{min} \gg \sum_{i}^{} \sum_{t}^{} \sum_{j}^{} c_{i,t} power_{i,t}^{(j)}$$

Or, on a

$$\begin{aligned} \sum_{i}^{} \sum_{t}^{} \sum_{j}^{} c_{i,t} power_{i,t}^{(j)} &\leq \sum_{i}^{} \sum_{t}^{} \sum_{j}^{} c_{i,t} max_{i,t} \\ &\leq S \sum_{i}^{} \sum_{t}^{} c_{i,t} max_{i,t} \end{aligned}$$

avec S le nombre de scénarios. Si l'on borne de telle manière que $c_{i,t} \leq c$ et $max_{i,t} < max$.

$$CQ_{min} \gg S \sum_{i}^{} \sum_{t}^{} c \times max$$

$$CQ_{min} \gg S \times T \times M \times c \times max$$

Cependant, Q_{min} est à priori inconnu. Par contre, cette quantité dépend de la puissance totale disponible, et, au pire, aura une valeur égale à $M \times max$. On a donc :

$$C \gg T \times S \times c$$

Délestage actif

On aimerait avoir un délestage à la période $t = 1$ avec une valeur de puissance délestée R, on aura donc l'objectif suivant :

$$\min R - Q^{-}_{va_1}$$

A partir du signe de la valeur de l'objectif final, on s'apercevra directement si la valeur est atteinte.

3.3.4 Contraintes

- **Bornes sur le niveau et la puissance :**
  $\forall i, t, j$:

$$min_{i,t} \leq level^{(j)}_{i,t} \leq max_{i,t}$$

$$l_{i,t}on^{(j)}_{i,t} \leq power^{(j)}_{i,t} \leq u_{i,t}on^{(j)}_{i,t}$$

  On notera que si les coefficients $a_{i,t}$ et $b_{i,t}$ sont nuls, il est alors inutile d'allumer la charge ce qui, par conséquent, impose $level^{(j)}_{i,t}$ et $on^{(j)}_{i,t}$ à 0.
- **Évolution du niveau :**
  $\forall i, t \geq \delta_i, j$:

$$level^{(j)}_{i,t+1} = level^{(j)}_{i,t} - out_{i,t} + \left(a_{i,t-\delta_i} power^{(j)}_{i,t-\delta_i} + b_{i,t-\delta_i} on^{(j)}_{i,t-\delta_i}\right)$$

  $\forall i, t < \delta_i, j$:

$$level^{(j)}_{i,t+1} = level^{(j)}_{i,t}$$

  Notons la présence d'une phase d'initialisation. Pour les $delta_i + 1$ premières périodes pour la charge i, on ne peut influer sur leur niveau étant donné que la puissance qui est associée à ces niveaux précède la référence de temps.
- **Réglage à la hausse :**
  $\forall j \in S_h, t \in \{\frac{j}{2} + 1, ..., \frac{j}{2} + N\}$:

$$\sum_i power^{(j)}_{i,t} = \sum_i power^{(j)}_{i,\frac{j}{2}} + Q^{(j)}_{va_t}$$

$$Q^{+}_{va_{\frac{j}{2}}} \leq Q^{(j)}_{va_t}$$

- **Réglage à la baisse :**
  $\forall j \in S_b, t \in \{\lfloor\frac{j}{2}\rfloor + 1, ..., \lfloor\frac{j}{2}\rfloor + N\}$:

$$\sum_i power^{(j)}_{i,t} = \sum_i power^{(j)}_{i,\lfloor\frac{j}{2}\rfloor} - Q^{(j)}_{va_t}$$

$$Q^{-}_{va_{\lfloor\frac{j}{2}\rfloor}} \leq Q^{(j)}_{va_t}$$

- **Quantités fixes :**
  $\forall t \leq T - N - 2 - \max_i \delta_i$:

$$Q^+_{fi} < Q^+_{va_t}$$

$$Q^-_{fi} < Q^-_{va_t}$$

  Expliquons l'intervalle de temps pour t qui correspond à celui pour lequel les quantités de réglages sont valides. Si l'on regarde l'équation d'évolution du niveau en prenant le temps maximal compris dans la modélisation, c'est à dire $T-1$, on a :

$$level^{(j)}_{i,T-1} = level^{(j)}_{i,T-2} - out_{i,T-2} + \left(a_{i,T-2-\delta_i} power^{(j)}_{i,T-2-\delta_i} + b_{i,T-2-\delta_i} on^{(j)}_{i,T-2-\delta_i}\right)$$

  Donc les puissances n'ont pas de niveaux associés si elles sont prises à un temps $\tau > T-2-\delta_i$. Concernant les quantités de réglage, celles-ci sont définies sur base de la somme des puissances. Pour connaître la quantité de réglage disponible au temps t, il faudra totaliser les puissances en $t, t+1, ...t+N$. Au mieux, on a la connaissance totale des puissances en $T-2-\max_i \delta_i$, ce qui permet de déterminer au maximum les quantités de réglage en $T-2-N-\max_i \delta_i$.
- **Variables communes aux scénarios :**
  $\forall i, t, j \neq S-1 : t \leq \lfloor \frac{j}{2} \rfloor$:

$$level^{(j)}_{i,t} = level^{(S-1)}_{i,t}$$

$$power^{(j)}_{i,t} = power^{(S-1)}_{i,t}$$

$$on^{(j)}_{i,t} = on^{(S-1)}_{i,t}$$

  Pour les périodes qui précèdent la demande de réglage, on demande à avoir un comportement identique. On prendra pour référence le dernier scénario où la demande de réglage est exécutée à la dernière période.
  Notons que ces contraintes mènent à une simplification : il n'est pas nécessaire d'instancier les variables du membre de gauche. Il suffit d'utiliser la variable du membre de droite en lieu et place dans toutes les autres équations. Ce faisant, la résolution grâce à l'algorithme du *branch and bound* gagnera énormément en rapidité vu que la taille du problème diminue significativement.

4 Un problème réalisable

Il va être discuté, dans ce chapitre, de la possibilité de déterminer aisément si le problème avec les données entrées est réalisable ou non. Cette partie, essentielle dans les algorithmes présentés par la suite, permet de ne pas se lancer dans la résolution du problème si celui-ci n'est pas réalisable. Nous allons voir que ceci peut-être accompli à l'aide de simples inégalités à tester.

Un résultat dérivé, mais non moins important, peut être obtenu : la possibilité d'améliorer les bornes sur le niveau minimal et maximal. Par *"améliorer"*, on entend augmenter la borne minimum et diminuer la borne maximum. De cette manière, on peut connaitre avec plus de précision les niveaux que peuvent prendre les réservoirs associés aux différentes charges.

Pour davantage de lisibilité, les indices sur les scénarios (j) et sur les charges (i) ont étés omis.

4.1 Contraintes à vérifier

4.1.1 Contraintes sur une période de temps

On établit aisément les contraintes suivantes provenant directement de la modélisation :

$$l_t \leq u_t$$
$$min_t \leq max_t$$

Celles-ci sont valables aussi bien pour toute période de temps que pour toute charge.

4.1.2 Contraintes sur deux périodes de temps consécutives

Nous allons voir maintenant quelles sont les conditions nécessaires pour qu'il y ait un état accessible en t connaissant l'état au temps $t-1$.

L'explication peut se faire sur base de la figure 4.1. A partir de l'état en t (en bleu), on peut soit éteindre la charge et arriver en t à un état déterminé (en noir). Soit, on alimente la charge pour arriver à une zone accessible (en vert) fonction de la puissance à laquelle on alimente la charge.

On comprend immédiatement que si les bornes sur le niveau sont toutes deux comprises entre la zone verte et le point noir, il n'y aura pas d'état réalisable au temps t. Notons que cette zone représente la partie accessible par la relaxation linéaire du problème si le coefficient b est nul.

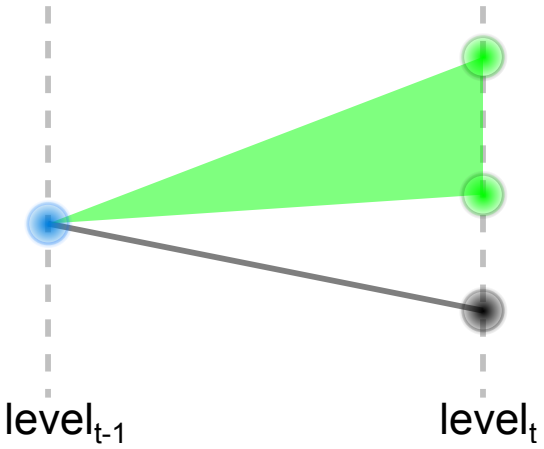

FIGURE 4.1 – Etats accessibles en t à partir de l'état en $t-1$. En bleu, le niveau de la charge en $t-1$. Si la charge n'est pas alimentée, son niveau va suivre la courbe en noir. Si, par contre, la charge est alimentée, son niveau sera donné par la partie en vert. Comme la puissance associée à la charge est comprise entre deux valeurs l et u, il est logique d'obtenir une plage de niveaux possibles.

Mathématiquement, cela peut s'exprimer de la manière suivante :

$$max_t \geq level_{t-1} + a_{t-1} * l_{t-1} + b_{t-1} - out_{t-1} \quad \text{ou} \quad min_t \leq level_{t-1} - out_{t-1}$$

Cependant, on n'a pas, a priori, la connaissance de l'état au temps $t-1$ avant la résolution proprement dite du problème. Il va donc falloir travailler avec les états possibles aux temps précédents. Ils sont logiquement compris entre la borne minimale et la borne maximale du niveau au temps $t-1$. L'adaptation est alors immédiate comme on peut le voir à la figure 4.2.

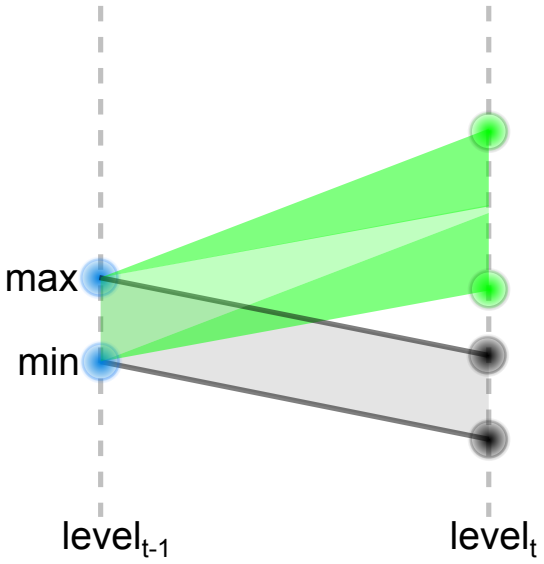

FIGURE 4.2 – Etats accessibles en t à partir des états possibles en $t-1$. L'ensemble des niveaux que peut prendre le réservoir est compris entre le minimum et le maximum. Au lieu d'avoir un seul niveau en $t-1$ comme à la figure 4.1, on obtient une plage de valeur.

Il ne reste plus qu'à transposer mathématiquement les nouvelles conditions, ce qui nous donne évidemment :

$$max_t \geq min_{t-1} + a_{t-1} * l_{t-1} + b_{t-1} - out_{t-1} \quad \text{ou} \quad min_t \leq max_{t-1} - out_{t-1}$$

Notons que le non-respect de cette condition peut mener à un problème dont la relaxation linéaire est réalisable car on peut "allumer à moitié la charge" ($on_t = 0.5$).

4.1.3 Contraintes sur plusieurs périodes de temps : plus petit maximum

A cela, il faut ajouter une vérification supplémentaire : un minimum trop élevé à un certain temps pourrait empêcher la possibilité de réaliser un maximum trop bas et inversement.

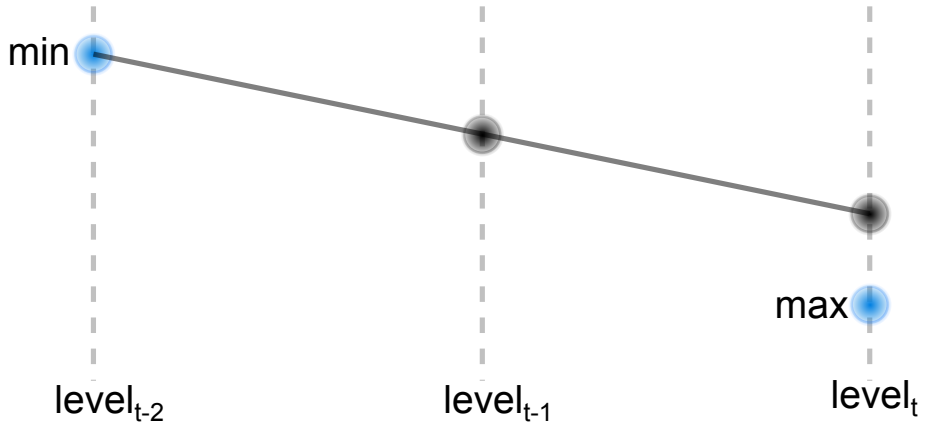

FIGURE 4.3 – Minimum trop élevé au temps $t-2$ ne permettant aucun état accessible au temps t. A partir du niveau initial en $t-2$, on obtient la courbe en noir des niveaux atteints en coupant l'alimentation de la charge.

Commençons par voir la borne inférieure d'un niveau maximum au temps T par rapport à un niveau minimum donné au temps $T-k$ avec $k = 0, ..., T-1$. Évidemment, nous prendrons le cas d'une alimentation totalement coupée pour déterminer le niveau maximal atteignable à partir de la valeur minimale du temps $T-k$.

$$\begin{aligned}
\text{level}_{T-k+1} &= \text{min}_{T-k} + 0 - \text{out}_{T-k} \\
\text{level}_{T-k+2} &= \text{min}_{T-k} - \text{out}_{T-k} - \text{out}_{T-k+1} \\
&\dots \\
\text{max}_T \geq \text{level}_T &= \text{min}_{T-k} - \sum_{\tau=T-k}^{T-1} \text{out}_\tau
\end{aligned}$$

Remarquons maintenant qu'il n'est pas nécessaire de vérifier, pour tous temps t, les minimums possibles dus aux maximums des temps 0 à $t-1$.

Comme on peut le voir à la figure 4.4, un minimum à une période $t-k$ plus haut que le plus petit maximum possible à la période $t-k$ dû à un minimum en $t-l$ avec $l > k$ est plus contraignant et peut donc devenir le nouveau plus petit maximum possible.

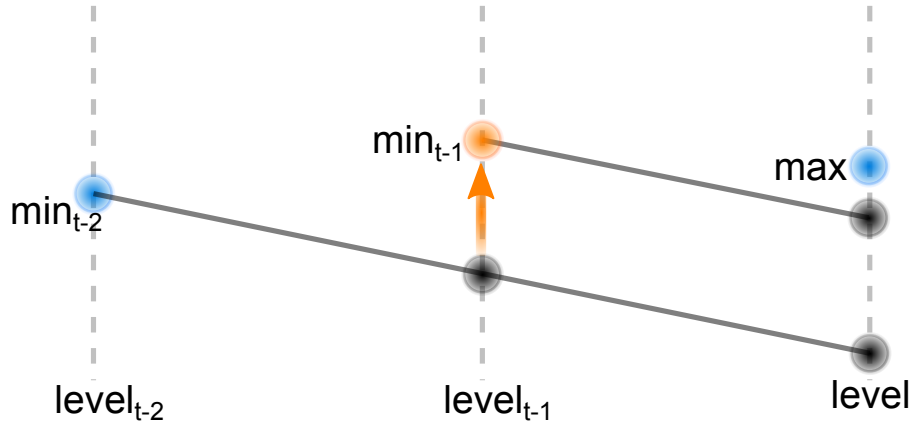

FIGURE 4.4 – Le minimum en période $t-1$ rehausse le plus petit maximum possible par rapport au minimum en $t-2$.

4.1.4 Contraintes sur plusieurs périodes de temps : plus grand minimum

Pour le cas du minimum, le raisonnement est identique mais ici, on délivrera à la charge la puissance maximale.

$$\min_T \leq \text{level}_T = \max_{T-k} + \sum_{\tau=T-k}^{T-1} (a_\tau u_\tau + b_\tau - out_\tau)$$

Cependant, il est cette fois nécessaire de prendre en compte la possibilité d'une saturation. En effet, une fois que l'on a activé la charge à sa puissance maximale jusqu'à atteindre le niveau maximal, nous n'aurons parfois plus d'autres choix que de couper l'alimentation de la charge aux périodes précédentes. Cette solution est illustrée à la figure 4.5 en bleu.

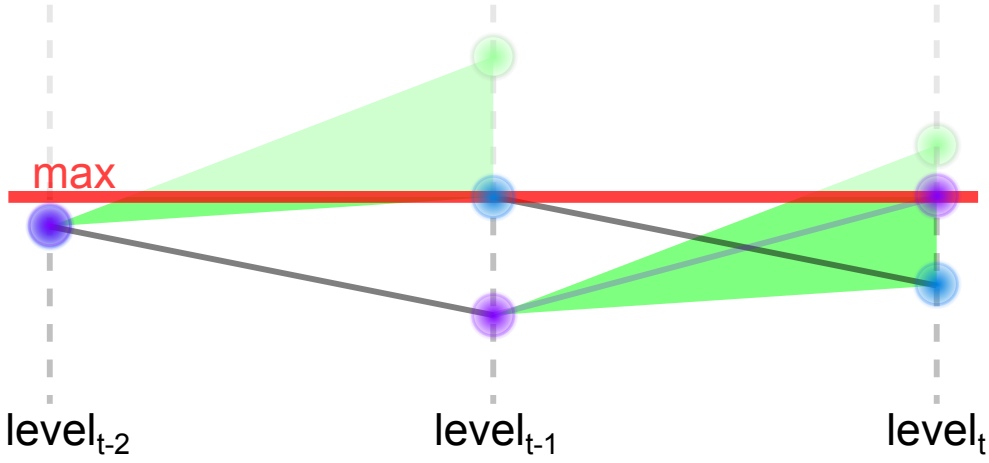

FIGURE 4.5 – L'existence d'un phénomène de saturation mal pris en compte (solution en bleu) rejetterait une solution réalisable (telle que celle en mauve).

Cependant, si l'on avait choisi de diminuer la puissance à cet instant pour l'augmenter plus tard, peut-être qu'à la période suivante, on aurait pu atteindre des niveaux supérieurs. Ce cas est également présenté à la figure 4.5 en mauve.

Prendre exactement cet effet en compte s'avérait être fort compliqué pour son apport réel alors qu'une solution simple se présentait : traiter ce problème comme un phénomène de saturation. C'est

à dire que, une fois le niveau maximum max_τ atteint, on spécifiera que le niveau maximum possible est égal à ce niveau maximum max_τ et ce, pour l'ensemble des périodes suivantes. Cependant, si plus tard, le maximum vient à augmenter subitement à une valeur supérieure à $\max_\tau$, on peut alors continuer à faire évoluer la borne vers le plus grand minimum possible à l'aide de la formule donnée au début de la section en partant de ce niveau $\max_\tau$.

4.2 Amélioration des bornes min et max

Vérifier la réalisabilité du problème est également une bonne occasion d'améliorer les bornes min et max pour celui-ci. Comme présenté juste après, nous avons déjà beaucoup d'informations à l'issue de la vérification de réalisabilité pour effectuer le renforcement.

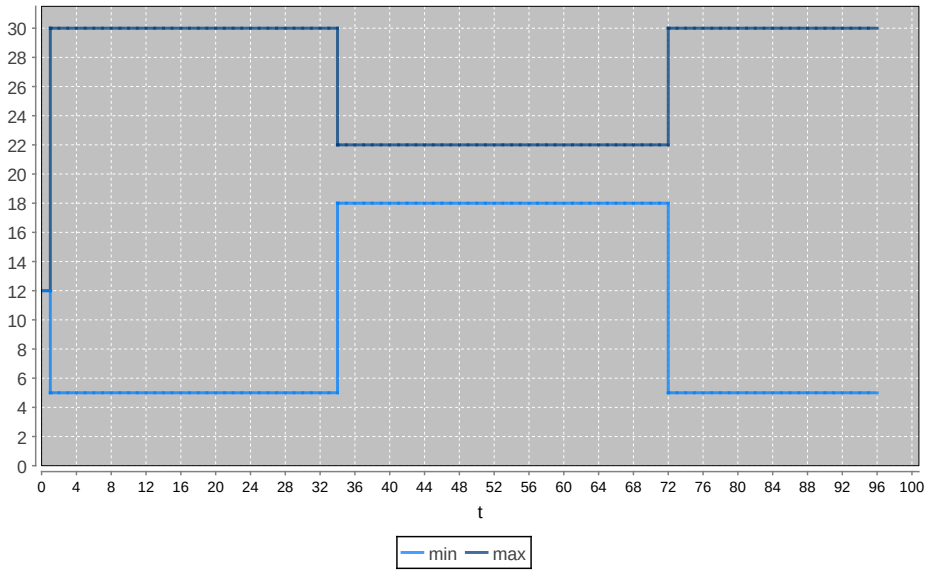

FIGURE 4.6 – Bornes initiales.

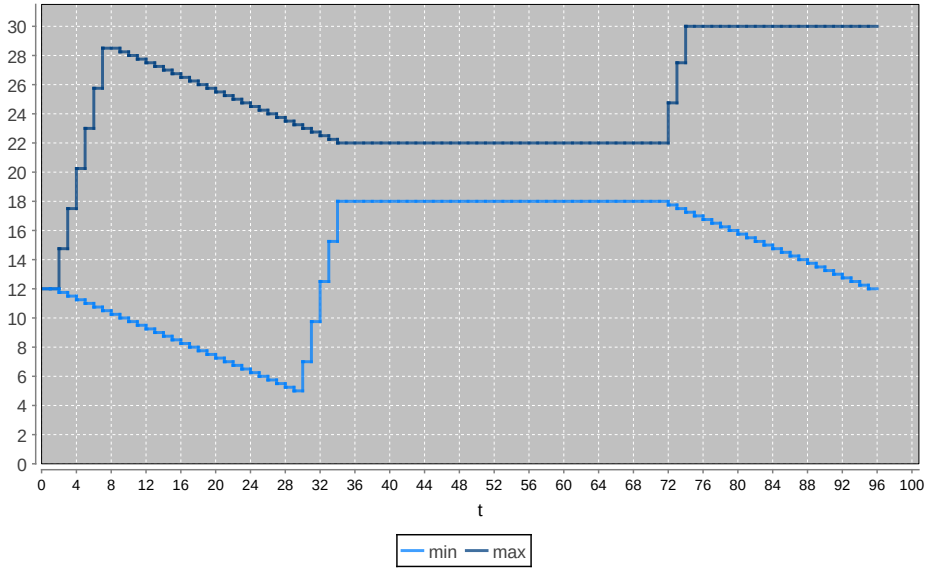

FIGURE 4.7 – Bornes améliorées.

L'idée ici est de ne pas se priver d'un résultat aisé à obtenir. De plus, cela nous permettra de connaitre avec plus de précision les niveaux réellement atteignables par nos *réservoirs génériques*. Finalement, cela permettra à l'algorithme de récupération de réalisabilité de se rendre compte plus tôt qu'une modification devra être effectuée pour obtenir la réalisabilité et par conséquent réduira la complexité du problème, mais ce point sera abordé dans l'explication de cet algorithme.

Si nous prenons le cas d'un chauffage dont les bornes sont illustrées à la figure 4.8, nous pouvons améliorer les bornes min et max jusqu'à arriver au résultat présenté à la figure 4.7.

4.2.1 A partir des minimums et maximums possibles

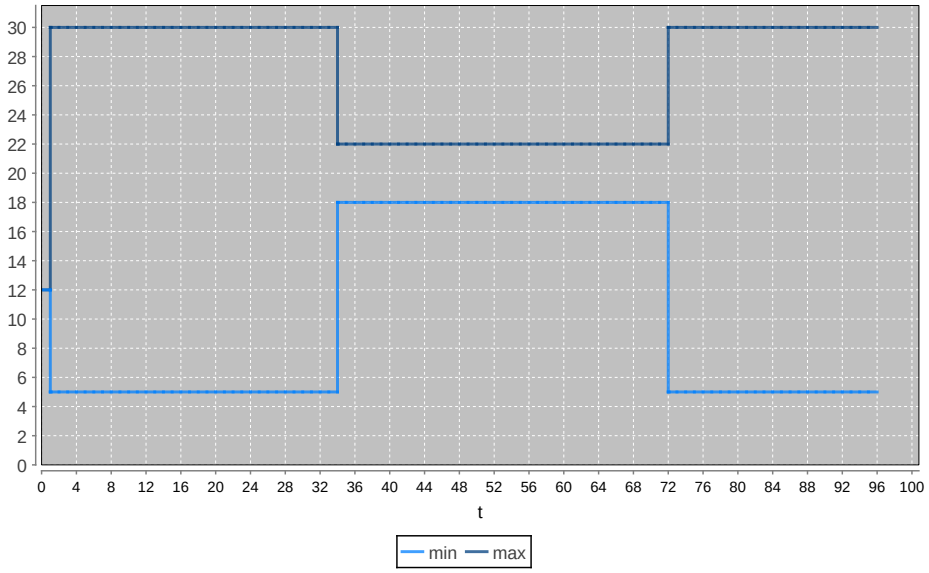

FIGURE 4.8 – Bornes initiales

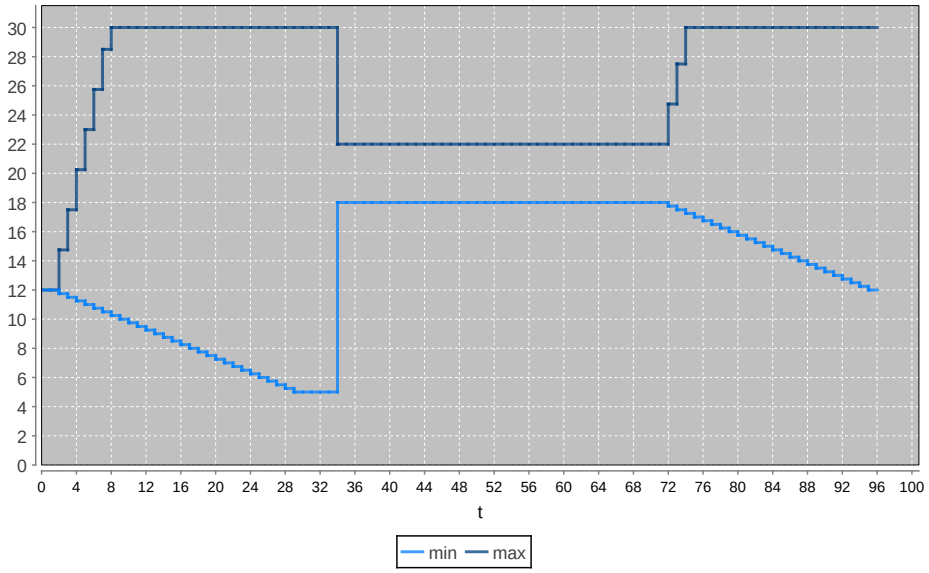

FIGURE 4.9 – Bornes améliorées

Nous avons précédemment parlé de maximums possibles comme étant les maximums les plus bas de sorte que le problème reste réalisable. Ceux-ci étaient obtenus à partir d'un minimum particulièrement haut et définissaient pour les périodes ultérieures, les niveaux minimums accessibles en coupant totalement l'alimentation de la charge pour toute période à venir. Il parait donc évident de prendre comme **nouveaux minimums** ces valeurs de **maximums possibles**.

Le raisonnement est totalement symétrique et par conséquent, nos **nouveaux maximums** seront donc les valeurs des **minimums possibles**.

Suite à cette explication, on se rend alors compte qu'à l'implémentation, il suffit de deux simples affectations pour passer d'une vérification de réalisabilité à une amélioration de bornes.

Les figures 4.8 et 4.9 permettent de visualiser l'effet de cette amélioration.

4.2.2 Conditions *a posteriori*

Avec à peine plus de difficultés, il est possible de passer du résultat précédent, illustré à la figure 4.9, à celui donné plus haut à la figure 4.7.

En effet, dans le cas d'un maximum très bas à une période t, le maximum en $t-1$ ne peut être bien plus haut. Identiquement, un minimum très haut en t permet de revoir à la hausse un minimum trop bas en $t-1$.

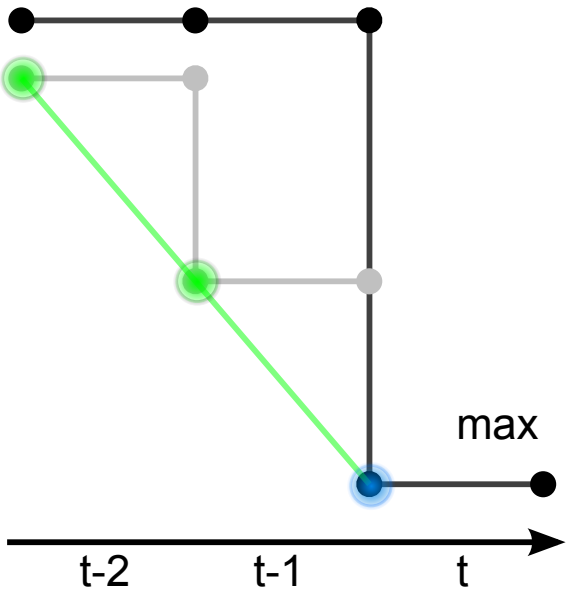

FIGURE 4.10 – Maximum *a posteriori*. Si l'on considère en t le niveau maximum, on obtient en vert les niveaux correspondant à une évolution en coupant l'alimentation de la charge aux temps précédents.

Étudions l'effet d'un maximum en t sur celui en $t-1$. Dans le pire des cas, on peut atteindre max_t à partir d'un niveau max^*_{t-1} en coupant l'alimentation de la charge. Par conséquent, on a :

$$max^*_{t-1} = max_t + out_{t-1}$$

Concernant le minimum, le raisonnement est identique, sauf que cette fois, on activera la charge à sa puissance maximale.

$$min^*_{t-1} = min_t + out_{t-1} - a_{t-1-\delta} * u_{t-1-\delta} - b_{t-1-\delta}$$

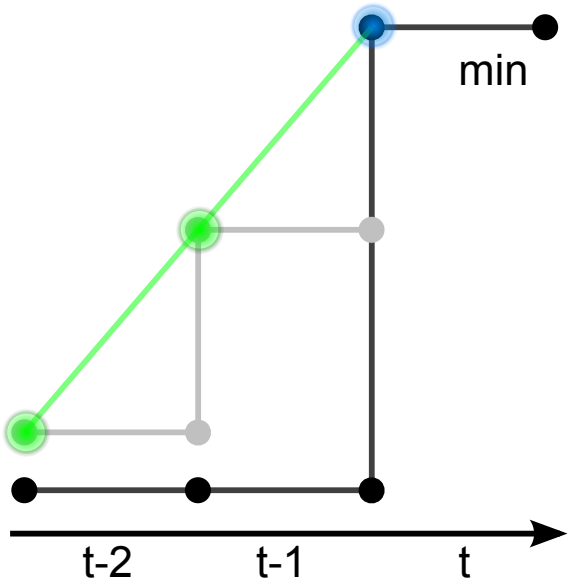

FIGURE 4.11 – Minimum *a posteriori*. Si l'on considère en t le niveau minimum, on obtient en vert les niveaux correspondant à une évolution en attribuant la puissance maximale à la charge aux temps précédents.

5 Algorithme de résolution par "niveau cible"

5.1 Présentation

Si l'on cherche la solution à l'aide d'un algorithme d'optimisation en nombre entiers classique, on obtient une solution d'une qualité indiscutable puisque optimale. Le temps pour la trouver par contre, lui, est conséquent, sans parler des ressources nécessaires pour utiliser l'algorithme du simplex avec un problème d'une taille acceptable pour notre application. On citera par exemple le fait que, pour la résolution d'un problème de 100 **charges** sur 48 périodes de temps, sur un ordinateur doté de 2 processeurs Intel® Core i7 quadri-cœurs cadencés à $3.33\ GHz$, aucune solution réalisable n'a pû être trouvée au bout de **45 minutes**.

Malheureusement, la complexité est exponentielle avec la taille du problème. En effet, l'algorithme du *branch and bound* va énumérer, dans le pire des cas, toutes les solutions associées aux différentes combinaisons de nos variables binaires, à savoir dans notre cas, la variable $on_{i,t}^{(j)}$. La complexité du problème est donc, ici, de $2^{\frac{M.T.S}{2}}$ avec M le nombre de charges, T le nombre de périodes et S le nombre de scénarios. La division par deux provient des contraintes de variables communes qui éliminent naturellement la moitié des variables. Si on n'élimine aucun scénario, on a donc $S = 2T$ et la complexité est alors de $2^{M.T^2}$.

Comme nous avons une idée de l'objectif à atteindre, nous pouvons penser à réaliser notre propre algorithme avec pour objectif d'approcher au maximum la solution optimale tout en prenant un temps considérablement plus faible.

5.2 Idée générale

L'idée générale est la suivante : nous allons tenter *"d'empiler"* les différentes charges à disposition afin d'obtenir une quantité de réglage globale.

Les étapes sont les suivantes et seront expliquées tour à tour ci-dessous :

1. Vérification de la réalisabilité et amélioration des bornes
2. Détermination du niveau global optimal
3. Détermination de la solution pour le dernier scénario
4. Détermination de la solution pour les autres scénarios
5. Calcul des quantités de réglages

5.3 Explication de l'algorithme

5.3.1 Vérification de la réalisabilité et amélioration des bornes

Cette partie a déjà été discutée précédemment. Pour rappel, on vérifie individuellement pour chaque charge que le niveau minimal et maximal est vraisemblable. C'est également l'occasion de renforcer ces bornes.

L'utilité du renforcement est d'avoir une idée réelle du véritable *"réservoir"* à notre disposition afin de travailler avec une information la plus exacte possible. Par exemple, pour des niveaux compris entre 0 et 100, si le minimum atteignable sur l'ensemble des périodes de temps étudiées est de 20, on aura un réservoir équivalent avec des niveaux compris entre 20 et 100.

5.3.2 Détermination du niveau global optimal

Une fois que l'on est assuré de la réalisabilité du problème, nous allons déterminer, pour chaque période temporelle, un coefficient traduisant notre volonté d'atteindre une plus grande quantité de réglage à la hausse par rapport à celle à la baisse, ou inversément.

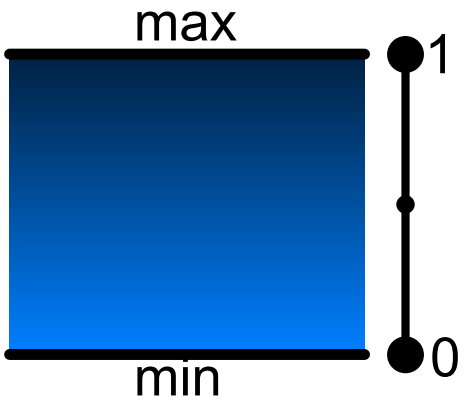

FIGURE 5.1 – Coefficient cible pour la détermination d'un niveau global optimal. Un nombre entre 0 et1 permet de traduire une volonté d'obtenir, pour chaque charge, un niveau proche respectivement du *minimum* ou du *maximum*.

Ce coefficient aura une valeur comprise entre 0 et 1. Dans le cas d'un coefficient valant 1, pour toute charge, nous tenterons d'atteindre le niveau maximal possible. Ainsi, un réglage à la baisse sera d'autant plus facile à réaliser et avec des quantités plus importantes. A l'inverse, un coefficient égal à 0 désigne la volonté d'atteindre le niveau minimum possible pour chaque charge et ainsi rendre plus aisé les possibilités de réglages à la hausse.

Nous noterons par la suite le coefficient pour la période t de la manière suivante : λ_t.

Dans un premier temps, nous choisirons de prendre la totalité des coefficients à 0.5. Le but de cette démarche est d'obtenir une flexibilité égale par rapport aux réglages à la hausse et à la baisse. Ce comportement sera raffiné par une recherche locale qui sera abordée par la suite.

5.3.3 Détermination de la solution pour le dernier scénario

C'est sans doute la partie la plus critique de l'algorithme : la détermination de la solution pour le dernier scénario. En effet, cette solution va servir de base jusqu'à la période de réglage pour tous les scénarios. Une fois cette période atteinte, on n'a que très peu de contrôle sur la suite des événements.

Par conséquent, il est important de bien choisir la solution puisqu'elle prédétermine les quantités de réglage disponibles.

On va donc avancer dans le temps en construisant la solution individuellement pour chaque charge. [1]

Pour chaque charge, nous allons viser un *niveau optimal* à atteindre basé sur les *coefficients cibles*.

Ce niveau optimal pour une charge i est calculé simplement à partir du coefficient λ_t de la manière suivante :

$$optimal_t = max_t.\lambda_{t-1-\delta_i} + min_t.(1 - \lambda_{t-1-\delta_i})$$

Comme le coefficient λ_t est compris entre 0 et 1, le niveau optimal sera donc forcément dans l'intervalle $[min; max]$. Il est logique d'aller chercher le coefficient à la période $t-1-\delta_i$. En effet, le niveau optimal est évidemment une grandeur en niveau alors que le coefficient λ_τ a été défini en terme de réglage, et donc de puissance. Il est donc tout à fait normal de voir apparaitre le temps $\tau = t-1-\delta_i$ qui fait le lien entre la période du niveau t et la période de la puissance qui lui correspond τ. Ce temps τ apparait plus clairement dans l'équation de *l'évolution du niveau* si on la réécrit pour obtenir le niveau du temps t :

$$\text{level}_{i,t}^{(j)} = \text{level}_{i,t-1}^{(j)} - \text{out}_{i,t-1} + \left(a_{i,t-1-\delta_i}\text{power}_{i,t-1-\delta_i}^{(j)} + b_{i,t-1-\delta_i}\text{on}_{i,t-1-\delta_i}^{(j)}\right)$$

Une fois ce niveau optimal déterminé pour la charge, l'algorithme va devoir décider à quelle puissance alimenter la charge. Les différents cas possibles sont présentés à la figure 5.2.

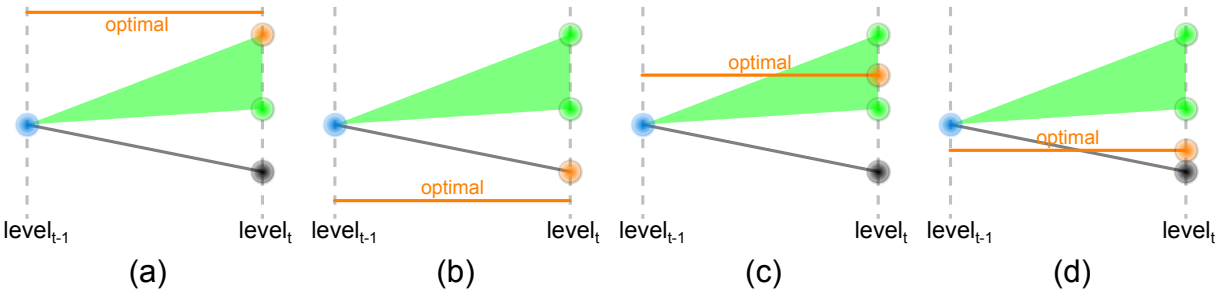

FIGURE 5.2 – Décision du niveau atteint par la charge au temps t en fonction du niveau optimal. En bleu, on retrouve le niveau en $t-1$. En noir, on observe le niveau en t si l'alimentation de la charge est coupée. Au vert correspond les niveaux accessibles en t si la charge est alimentée. La décision du niveau choisi est présentée en orange pour différents cas de *"niveaux optimums"*.

Si le niveau optimal est supérieur au niveau maximal (figure 5.2 - a), c'est à dire si l'on active la charge à sa puissance maximale, on choisira l'activation à cette puissance maximale. Si par contre, le niveau optimal ne peut être atteint même en coupant l'alimentation (figure 5.2 - b), on choisira de l'éteindre. Dans le cas où l'on peut atteindre ce niveau optimal, on alimentera la charge en conséquence (figure 5.2 - c). Le dernier cas (figure 5.2 - d) est celui où le niveau optimal est atteignable par la relaxation linéaire du problème. On choisira donc dans un premier temps d'alimenter la charge avec une puissance que l'on sait erronée.

5.3.4 Récupération de la réalisabilité

Dans les décisions précédentes, nous ne nous sommes pas préoccupés de la réalisabilité de la solution actuelle, un critère évidemment indispensable.

1. L'individualisation suggère évidemment une possibilité de *multi-threading*.

On distingue pour ce problème, deux types de réalisabilité :
- la réalisabilité de la puissance
- la réalisabilité du niveau

.

La première a été récupérée de manière assez simple. Si la charge a été choisie *"éteinte"*, on s'assure que la puissance associée est bien nulle. Si elle a été choisie *"allumée"*, on vérifiera juste que la puissance est entre les bornes qui lui sont associées. C'est à dire que pour une puissance supérieure à la puissance maximale, on lui assignera la puissance maximale et pour une puissance inférieure à la puissance minimale, on lui assignera celle-ci.

Il faut maintenant assurer que le niveau à la période de temps considérée est compris entre ses deux bornes min_t et max_t. Pour cela, une méthode récursive a été développée.

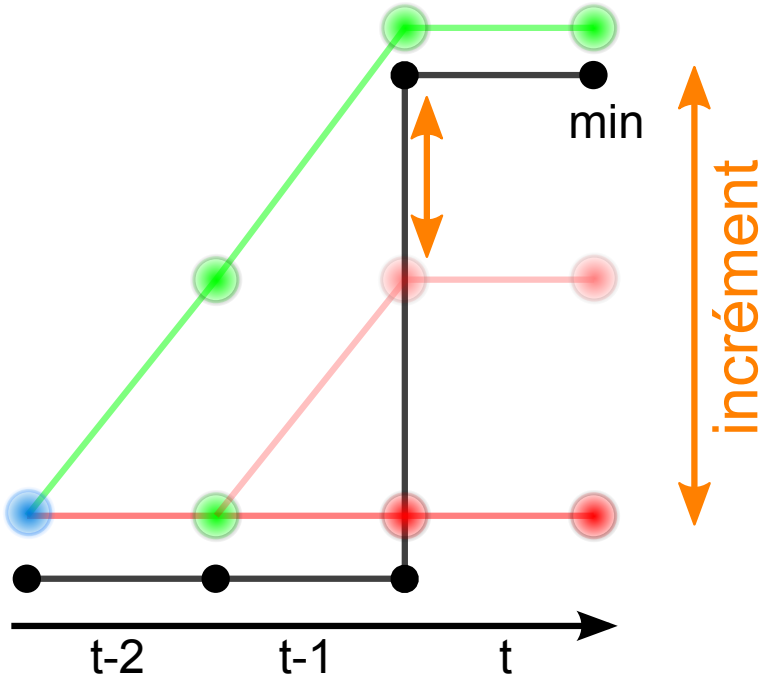

FIGURE 5.3 – Récupération de la réalisabilité dans le cas d'un minimum violé.

Étudions le cas d'un niveau optimal calculé inférieur au minimum. Regardons d'abord sur l'exemple donné à la figure 5.3. La solution initiale était valide en $t-2$ et $t-1$. Cependant, la solution en t, elle, donne lieu à un niveau en dessous du niveau minimal d'un certain **incrément** (en orange). L'algorithme va tout d'abord activer la charge en $t-1$ et ainsi diminuer l'incrément à récupérer. Cependant, ce n'est pas suffisant pour récupérer la réalisabilité, on va donc regarder pour une activation en $t-2$ afin de récupérer ce nouvel incrément.

Concrètement, l'algorithme pour récupérer la réalisabilité pour un niveau trop bas est le suivant :

```
def getLevelIncrement(t, increment) :
  # Definir la puissance adaptee
  p=puissanceMaximalePossible(t)
  l=niveauAssocie(t,p)
  if l-level(t) > increment :
    p=puissanceNecestsaire(t,increment)
    l=niveauAssocie(t,p);

  # Actualisation
  increment-= l-level(t)
```

```
  level(t)=l
  puissance(t-1-d)=p # avec d le delai

  # Appel recursif
  if (increment > 0)
    getLevelIncrement(t-1, increment)
    miseAJourNiveau();
```

Listing 5.1 – Récupération de la réalisabilité par un incrément de niveau.

On a évidemment un raisonnement identique pour un niveau supérieur à un maximum donné. Cela conduit donc à un algorithme similaire :

```
def getLevelDecrement(t, increment) :
  # Definir la puissance adaptee
  p=puissanceMinimalePossible(t)
  l=niveauAssocie(t,p)
  if level(t)-l > increment :
    p=puissanceNecessaire(t,increment)
    l=niveauAssocie(p);

  # Actualisation
  increment-= level(t)-l
  level(t)=l
  puissance(t-1-d)=p # avec d le delai

  # Appel recursif
  if (increment > 0)
    getLevelDecrement(t-1, increment)
    miseAJourNiveau();
```

Listing 5.2 – Récupération de la réalisabilité par un décrément de niveau.

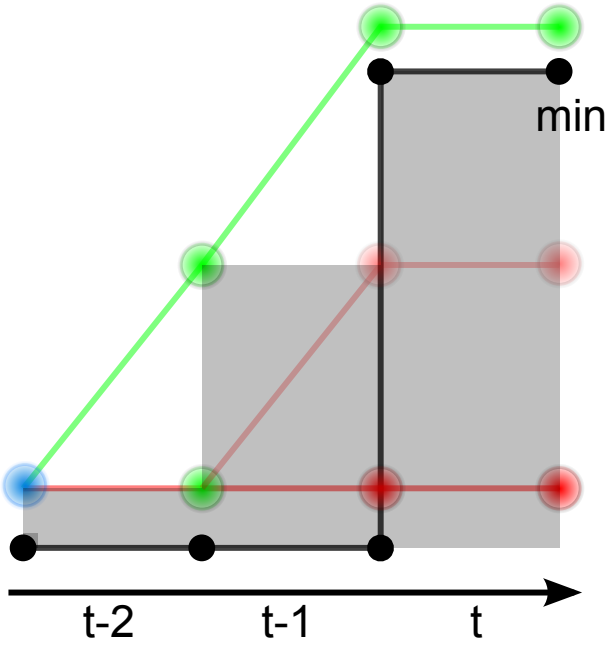

FIGURE 5.4 – Récupération de la réalisabilité dans le cas d'un minimum violé. L'effet de l'amélioration des bornes est présenté en gris. Grâce à cette amélioration des bornes, les solutions en rouge seront invalidées directement au lieu de l'être en t.

Lors de la vérification pour savoir si le problème est réalisable, une amélioration des bornes est effectuée. Si l'on observe le résultat de l'amélioration en gris à la figure 5.4 par rapport à la figure 5.3, on remarque que les solutions en rouge ne seront déjà pas possible en $t-1$ et le problème y sera donc réglé au lieu de le laisser se propager en $t-2$.

5.3.5 Détermination de la solution pour les autres scénarios

Une fois la solution pour le dernier scénario connue, on peut construire celle pour les autres scénarios de la manière suivante. Tout d'abord, on fixe les variables communes avec le dernier scénario, à savoir $on_{i,t,j}$, $power_{i,t,j}$ et $level_{i,t,j}$ pour $t \leq \lfloor \frac{j}{2} \rfloor$.

Ensuite, charge par charge, on distingue trois cas basés sur la variable $\tau = t - 1 - \delta$ qui lie le niveau du temps t avec la puissance qui lui correspond.

Pour $\tau < \lfloor \frac{j}{2} \rfloor$, la puissance de la charge associée au niveau t est déjà fixée et il suffit donc de calculer le niveau qui lui correspond.

La partie la plus importante pour la solution se passe pour $\lfloor \frac{j}{2} \rfloor < \tau \leq \lfloor \frac{j}{2} \rfloor$. En effet, c'est la puissance consommée dans ces intervalles de temps qui va déterminer la quantité de réglage disponible. Dans le cas d'un réglage à la hausse, on activera a priori la charge à sa puissance maximale et pour un réglage à la baisse, on choisira l'extinction. Si nécessaire, il faudra naturellement récupérer la réalisabilité de la même manière qu'expliqué précédemment.

Et enfin, pour $\tau \geq \lfloor \frac{j}{2} \rfloor$, nous ne pouvons améliorer la valeur de la fonction objectif qu'en activant les charges selon de leur coefficient de priorité. Si celui-ci est positif, on préfèrera activer la charge sinon, on l'éteindra. Finalement, nous rectifierons la solution pour atteindre la réalisabilité comme à notre habitude.

5.3.6 Calcul des quantités de réglages

Il s'agit ici d'une simple application des formules données dans le modèle. On calcule tout d'abord la quantité de réglage pour tous les scénarios à partir de la puissance totale juste avant la période de réglage servant alors de puissance de référence. Ensuite on regarde les puissances totales aux temps suivant la demande de réglage et prenons le minimum. En prenant la différence entre cette grandeur et la valeur de référence, on obtient la quantité de réglage pour cette période. Pour trouver la quantité de réglage fixe, il suffit alors de regarder le minimum sur l'ensemble des scénarios.

5.4 Complexité

Effectuons le calcul de la complexité de cet algorithme.

Celui-ci débute avec l'algorithme de vérification de réalisabilité, responsable également de l'amélioration des bornes. Celui-ci analyse chaque charge indépendamment et parcourt toutes les périodes dans un sens puis dans l'autre. Sa complexité est donc de $2.M.T$.

Ensuite, il y a la recherche de la solution du dernier scénario, et ce, pour tout temps et charges. Et pour chacun des choix en terme de puissance, il faut utiliser l'algorithme de récupération de réalisabilité. Celui-ci démarre du temps t et, si nécessaire, change les décisions prises aux temps précédents pour rendre la solution réalisable. Dans le meilleur des cas, la complexité est de 1 et dans le pire, $t-1$. On bornera celle-ci par T. Pour la recherche du dernier scénario, on a donc la complexité $M.T^2$.

Finalement, la méthode de résolution étend le choix du dernier scénario à tous les autres. Encore une fois, le calcul le plus lourd est, dans le pire des cas, celui de la récupération de la réalisabilité. On n'est donc pas surpris d'obtenir, pour cette partie, la complexité $(S-1).M.T^2$.

En définitive, à notre algorithme correspond une complexité de l'ordre de $S.M.T^2$ dans le pire des cas et dans le meilleur, de $S.M.T$.

5.5 Conclusion

Cette méthode permet d'obtenir une solution réalisable extrêmement rapidement mais, malheureusement, la qualité de celle-ci est médiocre. Dans la plupart des cas, les quantités de réglages maximales sont souvent négatives pour certaines périodes de temps. Visiblement, il va falloir trouver beaucoup mieux. Cela ne veut pas dire que tout ceci a été réalisé en vain, par la suite beaucoup sera réutilisé dans les algorithmes suivants.

La plus grande difficulté jusqu'à présent a été de concevoir l'algorithme de récupération de réalisabilité. Celui-ci sera utilisé dans toutes les méthodes qui vont suivre.

5.6 Amélioration par recherche locale

5.6.1 Introduction

Avec l'algorithme précédent, nous trouvions une solution rapidement mais celle-ci reste loin d'être optimale. Dans cette partie, nous nous concentrons sur la possibilité d'améliorer quelque peu cette solution en partant de la base établie et en effectuant une recherche locale.

Une solution fournie grâce aux concepts expliqués précédemment est donnée à la figure 5.5. On y repère la présence de creux importants entachant considérablement la quantité de réglage fixe. Remarquons également que, en général, ces creux dans la quantité de réglage sont accompagnés à la même période par un pic dans l'autre type de réglage.

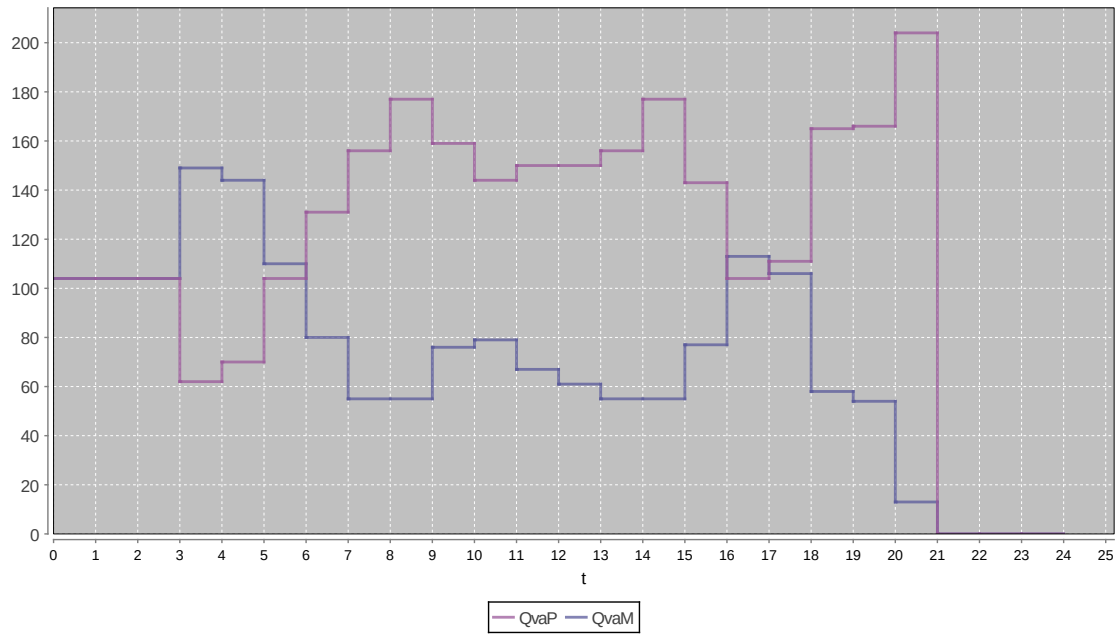

FIGURE 5.5 – Solution donnée par l'*algorithme de résolution par niveau cible* sans recherche locale. On y repère la présence de creux importants entachant considérablement la quantité de réglage fixe.

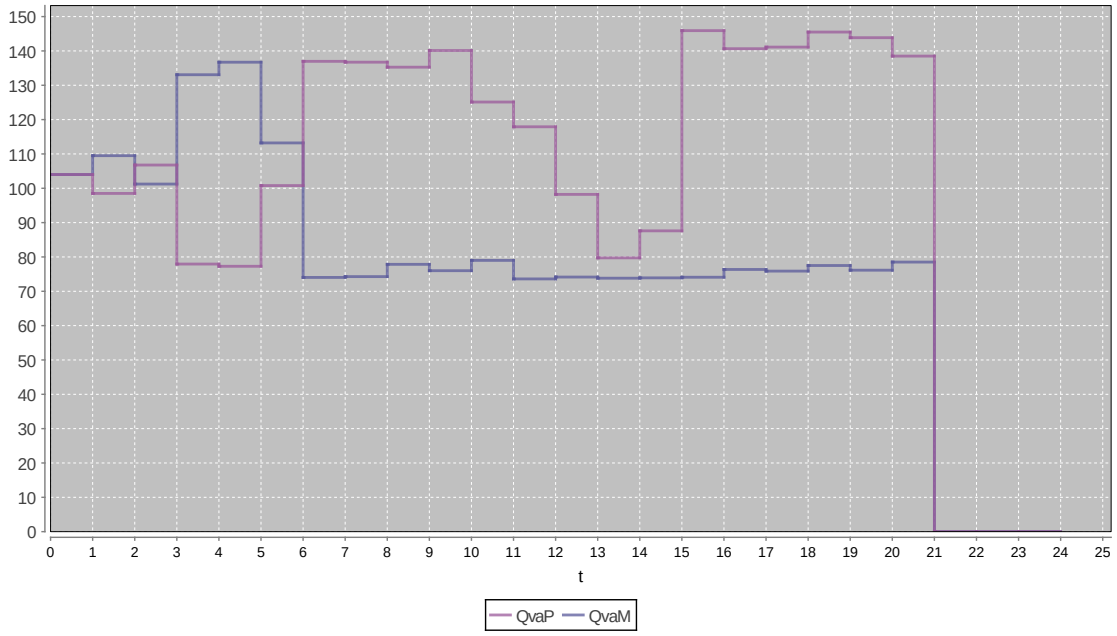

FIGURE 5.6 – Solution apportée par l'amélioration par recherche locale pour le même problème qu'à la figure 5.5

Prenons l'exemple d'un réglage à la baisse trop faible à une période avec, à l'inverse, un réglage à la hausse très important. Nous allons tenter de revoir la solution afin d'arriver à un juste milieu pour les deux quantités et ainsi tenter d'augmenter les quantités de réglage fixes.

5.6.2 Implémentation

Le système se base sur plusieurs itérations de l'algorithme de base en faisant varier les valeurs des *coefficients cibles pour la détermination d'un niveau global optimal*.

Pour rappel, ces coefficients d'une valeur entre 0 et 1 traduisaient la volonté d'atteindre plutôt le niveau minimal ou le niveau maximal. Réfléchissons correctement à ce que veut dire une carence en réglage à la hausse en t : cela signifie que dans les périodes suivant la périodes t, il n'a pas été possible d'augmenter la puissance significativement.

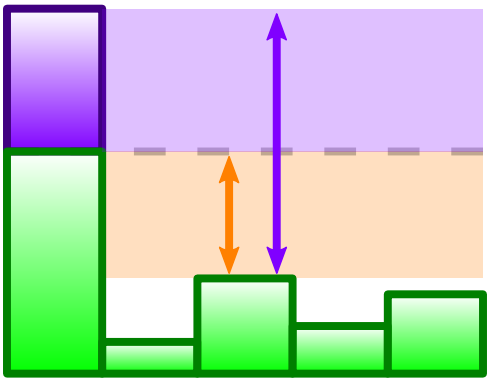

FIGURE 5.7 – Illustration des puissances pour un réglage à la baisse. La hauteur en orange correspond à la quantité de réglage à la baisse obtenue. En changeant le coefficient cible, on espère augmenter la puissance de référence de l'incrément en mauve et ainsi augmenter la quantité de réglage obtenue.

La puissance totale du parc de charges en t sert de référence pour déterminer la quantité de réglage disponible. La première idée va donc être de diminuer la puissance totale à la référence pour que l'augmentation aux périodes suivantes soit plus marquée. On va donc tenter d'**augmenter** le *coefficient cible* en t. Une fois que l'on aura manipulé au maximum la référence, c'est à dire saturé le coefficient correspondant, on modifiera aux périodes précédentes. Par contre, cette fois, nous diminuerons le coefficient. Évidemment, le raisonnement est symétrique pour les réglages à la baisse.

Algorithme général

Nous allons effectuer quelques itérations de l'algorithme de base en gardant la meilleure solution. Après chaque résolution par l'algorithme de base, nous allons ajuster les coefficients cibles. Nous continuerons jusqu'à ne plus rencontrer de meilleures solutions sur une longue durée. Ceci est modélisé par la variable *maxIterationObjectiveLess* dans le pseudo-code résumant l'algorithme donné ci-dessous.

```
def solve() :
  it=0;
  itObjectiveLess=0;
  best=-Inf;
  while it < maxIteration && itObjectiveLess < maxIterationObjectiveLess :
    # Calcul de la solution
    objectif=SimpleSolver.solve();
    if objectif > best :
      best=objectif;
      itObjectiveLess=0;
      # !!! garder la nouvelle solution
    else :
      ++itObjectiveLess
    ++it

    # Calcul coefficients
    ajusterCoefficients()
```

Listing 5.3 – Amélioration par recherche locale.

Ajustement des coefficients

Nous allons ici jouer sur les deux types de réglages. Tout d'abord, nous rechercherons les minimums entachant la quantité de réglage fixe. Afin d'aider à la compréhension, un pseudo-code est donné ci-dessous.

```
def ajusterCoefficients() :
  # Recherches des minimums
  minMTime=rechercheMinimumReglageBaisse()
  minPTime=rechercheMinimumReglageHausse()
  if minPTime == minMTime # l'ajustement ne peut favoriser l'un ou l'autre
    return

  # Ajout a la liste des periodes critiques
  critiquesM.add(minMTime)
  critiquesP.add(minPTime)

  # Reglage a la hausse : recherche de la periode a regler
  t=minPIndex
  while t >= 0 && ( estSature(coefficients[t]) || critiquesM.contient(t) ) :
    --t
```

```
# Reglage a la hausse : ajustement
if t >= 0 :
  ajustement=gamma
  if QvaM[t] > 0 :
    ajustement=gamma*(QvaM[t]-QfiM)/QvaM[t]

  if t == minPIndex :
    coefficients[t]=saturer(coefficients[t] - ajustement)
  else
    coefficients[t]=saturer(coefficients[t] + ajustement)

# Reglage a la baisse : recherche de la periode a regler
t=minMIndex
while t >= 0 && (estSature(coefficients[t]) || critiquesP.contient(t) ) :
  --t

# Reglage a la baisse : ajustement
if t >= 0 :
  ajustement=gamma
  if QvaP[t] > 0 :
    ajustement=gamma*(QvaP[t]-QfiP)/QvaP[t]

  if t == minPIndex :
    coefficients[t]=saturer(coefficients[t] + ajustement)
  else
    coefficients[t]=saturer(coefficients[t] - ajustement)
```

Listing 5.4 – Ajustement des coefficients

Cette partie peut être divisée en quelques étapes :

1. Recherche des périodes contenant les minimums de quantités de réglages à la hausse et à la baisse.
2. Ajout des périodes trouvées à la liste des périodes critiques
3. Pour les deux types de réglages :
   - Recherche de la période à régler
   - Ajustement du coefficient

Tout d'abord, il s'agit de rechercher la période où survient le plus faible résultat dans les différentes quantités de réglages. C'est sur celle-ci que l'on va se concentrer afin de tenter de remonter la quantité de réglage fixe donnée en solution.

Une fois que l'on connait les périodes problématiques, nous allons les ajouter à une liste, différente pour chaque type de réglage, qui contiendra toutes les périodes ayant posé problème au cours des itérations précédentes. Il serait mal avisé de modifier un coefficient pour un réglage à la hausse alors que celui-ci a été ajusté par le passé car étant problématique pour un réglage à la baisse.

Ensuite, nous allons rechercher quel coefficient modifier pour arriver à améliorer la quantité de réglage disponible à une période t. Il parait évident que si le coefficient à une période est à 0, il sera impossible de le diminuer davantage. De la même manière, un coefficient à 1 ne pourra être augmenté. Dans ce genre de cas, on choisira d'agir à une période précédente.

Finalement, il ne reste plus qu'à ajuster le coefficient à la période trouvée de manière intelligente. Mais de quelle quantité allons-nous modifier ce coefficient borné entre 0 et 1 ? Il faut trouver une grandeur qui soit l'image de la quantité de réglage disponible dans l'autre mode de réglage, qui est, en général, spécialement importante. On a donc pour un réglage à la hausse à améliorer au temps t :

$$\lambda_t^* = \lambda_t - \gamma.\frac{Q_{va_t}^- - Q_{fi}^-}{Q_{va_t}^-}$$

Et pour un réglage à la baisse à améliorer au temps t :

$$\lambda_t^* = \lambda_t + \gamma.\frac{Q_{va_t}^+ - Q_{fi}^+}{Q_{va_t}^+}$$

Et on assure par saturation que le nouveau coefficient λ_t^* est compris entre 0 et 1. γ est un paramètre d'amortissement, compris dans l'intervalle $]0;1[$, permettant d'adoucir la variation de λ_t^*.

Cependant, si l'on considère un réglage à la hausse à améliorer au temps t en modifiant le coefficient au temps $\tau < t$, il est alors nécessaire de réaliser l'ajustement inverse. On préfèrera diminuer le coefficient afin de favoriser une diminution de la puissance en t permettant ainsi une activation plus aisée en $t+1, ..., t+N$. Le raisonnement est symétrique pour un réglage à la baisse.

5.6.3 Conclusion

En quelques itérations de l'algorithme de base, on peut maintenant améliorer significativement la solution sans augmenter la complexité de la résolution, restant, dans le pire des cas, à $k.S.M.T^2$ où k est le nombre d'itérations que l'on va effectuer. Un avantage de cette recherche locale encore non précisé est que, une fois la première solution obtenue, ce qui est assez rapide, l'algorithme peut être arrêté à tout moment puisque une solution réalisable a déjà été obtenue.

Nous verrons, dans les tests, la qualité obtenue, mais avant, présentons d'autres manière de résoudre ce problème.

6 Algorithme de résolution par "puissance cible"

6.1 Idée générale

Précédemment, nous avions tenté de maximiser la quantité de réglage en ajustant des coefficients basés sur un niveau de réservoir cible. Il est maintenant temps d'essayer une solution différente. Cette fois, nous nous concentrerons sur une **puissance de référence cible**.

Évidemment, nous nous baserons sur ce qui a été développé précédemment, tout particulièrement l'algorithme de récupération de réalisabilité.

6.2 Notion de puissance de référence totale

Dans l'élaboration de l'algorithme précédent, nous avons remarqué que la construction de la solution du dernier scénario était la partie critique. La solution pour les autres scénarios découlaient tout naturellement des choix pour le dernier scénario avec très peu de possibilités d'actions par la suite. Voyons en quoi cet état de fait est lié à ce que nous appellerons une **puissance de référence**.

Pour rappel, les équations définissant la quantité des différents types de réglages sont données par :

- **Réglage à la hausse :**
  $\forall j \in S_h, t \in \{\frac{j}{2}+1, ..., \frac{j}{2}+N\}$:

$$\sum_i power_{i,t}^{(j)} = \sum_i power_{i,\lfloor \frac{j}{2} \rfloor}^{(j)} + Q_{va_t}^{(j)}$$

$$Q_{va_{\lfloor \frac{j}{2} \rfloor}}^{+} \leq Q_{va_t}^{(j)}$$

- **Réglage à la baisse :**
  $\forall j \in S_b, t \in \{\lfloor \frac{j}{2} \rfloor + 1, ..., \lfloor \frac{j}{2} \rfloor + N\}$:

$$\sum_i power_{i,t}^{(j)} = \sum_i power_{i,\lfloor \frac{j}{2} \rfloor}^{(j)} - Q_{va_t}^{(j)}$$

$$Q_{va_{\lfloor \frac{j}{2} \rfloor}}^{-} \leq Q_{va_t}^{(j)}$$

On nommera la grandeur "**puissance de référence totale**" apparaissant dans les équations ci-dessus comme étant :

$$P_{ref} = \sum_i power^{(j)}_{i,\lfloor \frac{j}{2} \rfloor}$$

Afin d'obtenir le même comportement pour tous les scénarios aux temps précédant la demande de réglage, nous avions fixé des variables communes. Ainsi, la quantité donnée par $\sum_i power^{(s)}_{i,\lfloor \frac{j}{2} \rfloor}$ est identique pour tous les scénarios $s \geq j$. Cela implique notamment que cette somme de puissances peut être définie uniquement à l'aide du dernier scénario.

6.3 Vers une puissance de référence totale cible

6.3.1 Facteur de référence

Discutons maintenant de la recherche proprement dite d'une solution. L'algorithme est présenté par le pseudo-code 6.1. Une fois le niveau initial fixé, on va considérer tour à tour toutes les périodes de temps. Pour chacune d'entre elles, on commence par trier les charges selon un certain ordre expliqué un peu plus loin. Ensuite, on va considérer ces charges dans l'ordre et les activer jusqu'à arriver à une **puissance de référence cible**. Une fois ce niveau dépassé, on choisira d'éteindre les charges. Évidemment, ces décisions ne permettent pas d'assurer la réalisabilité. Il faudra donc utiliser l'algorithme développé précédemment pour cette tâche.

A l'issue des ces quelques instructions, nous obtenons la solution pour le dernier scénario. Il ne reste plus qu'à étendre les résultats aux autres scénarios. Cela peut être effectué de la même manière que pour l'algorithme précédent en tentant d'activer toutes les charges pour un réglage à la hausse ou en tentant de toutes les éteindre pour un réglage à la baisse. Finalement, il ne reste plus qu'à calculer le résultat en terme de quantité de réglage.

```
# Niveau initial
for i in range(M) :
  level[i][0][j]=initial[i][0]

# Dernier scenario
for t in range(T) :
  puissanceDeReferenceTotale[t] = 0

  # Tri des charges
  chargesTriees = trierCharges()

  # Activation
  for i in range(M) :
    index=chargesTriees[i];

    if puissanceDeReferenceTotale[t] < puissanceTotalePossible[t]*facteurReference[t] :
      power[index][t][S-1]=u[index][t]
      on[index][t][S-1]=1
    else :
      power[index][t][S-1]=0
      on[index][t][S-1]=0

  # Realisabilite
  if t > 0 :
    recuperationRealisabilite(index, t, S-1)

# Resultat
solutionAutresScenarios()
```

```
calculQuantitesObtenues()
```

Listing 6.1 – Algorithme de recherche d'un niveau de référence cible

Finalement, on effectue donc un seuillage à l'aide d'une *puissance de référence totale cible* définie comme étant une fraction de la *puissance totale disponible* :

$$P_{ref_t} = \omega_t P_{totale_t}$$

$$\sum_i power_{i,t}^{(j)} = \omega_t \sum_i u_{i,t}$$

Évidemment, à l'issue de cet algorithme, notre puissance de référence pourrait avoir une puissance de référence différente de celle attendue. En effet, si l'on tente de couper les charges une fois le seuil atteint, certaines devront quand même être allumées pour avoir une solution respectant les contraintes sur le niveau de la charge.

6.3.2 Méthode de tri des charges

Une petite zone de flou a été laissée dans l'explication précédente : le critère de tri des charges. Nous pourrions nous contenter de prendre les charges dans l'ordre où celles-ci apparaissent mais évidemment, une décision mieux réfléchie semble être une idée raisonnable.

A chaque charge, nous allons attribuer une valeur. Plus celle-ci sera grande, plus la charge aura tendance à être activée. Par contre, plus cette valeur s'approchera de 0, plus la charge risquera d'être désactivée.

Tout d'abord, on peut mettre de côté les charges qui ne peuvent être activées, c'est à dire telles que $u_{i,t} = 0$. Ceci est logique étant donné que ces charges ne contribueront forcément pas à l'augmentation de la puissance de référence. Par conséquent, on leur attribuera une valeur égale à 0.

Ensuite, on mettra de côté les charges dont l'activation est certaine pour avoir une solution réalisable, c'est à dire telles que si l'on ne les alimente pas, le niveau qui va correspondre sera inférieur au minimum. On attribuera à ces charges une valeur extrêmement élevée.

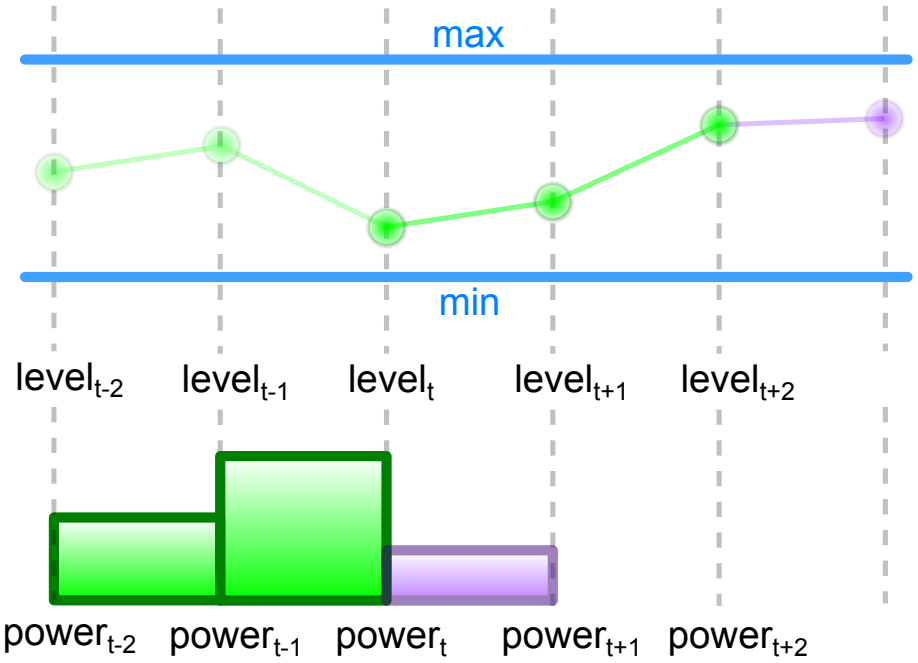

FIGURE 6.1 – État de la solution lors de la détermination de la puissance en t. On y voit en mauve la puissance à déterminer et le niveau qui y correspond en $\tau = t + 1 + \delta_i$ avec ici $\delta_i = 1$. En vert, on trouve les puissances déjà définies et les niveaux correspondants.

Si, par contre, avec une puissance $power_{i,t}^{(S-1)} = l_{i,t}$, le niveau correspondant est supérieur au niveau maximum, la charge devra être éteinte et aura donc une valeur de 0.

Une fois les cas extrêmes mis de côté, on peut se préoccuper du reste des charges. Définissons une distance par rapport au niveau maximum dans le cas d'une activation à la puissance maximale :

$$d^+ = max_{i,t} - \left(level_{i,t+\delta_i}^{(S-1)} - out_{i,t+\delta_i} + a_{i,t}.u_{i,t} + b_{i,t}\right)$$

et une distance par rapport au niveau minimum dans le cas d'une puissance nulle :

$$d^- = level_{i,t+\delta_i}^{(S-1)} - out_{i,t+\delta_i} - min_{i,t}$$

Empiriquement, la valeur obtenue donnant les meilleurs résultats est la suivante :

$$10000 + \frac{d^- - d^+}{\max_{i,t} - \min i, t}$$

La constante 10000 est présente uniquement afin de distinguer nettement les valeurs à 0 dues à une impossibilité d'activation des valeurs dues à des distances proches l'une de l'autre.

6.4 Adaptation de la puissance de référence cible

Une fois de plus, on a la possibilité d'améliorer aisément le résultat donné par l'algorithme précédent en ajustant la *puissance de référence totale cible* via la modification du facteur de référence ω_t. Nous allons donc utiliser exactement le même système que pour la recherche locale dans l'algorithme précédent ; c'est à dire garder en mémoire la meilleure solution, ajuster le facteur ω_t puis regarder si cela implique une meilleure solution.

Le principe de l'ajustement est encore une fois très simple. Tout d'abord, on recherche le minimum de quantité de réglage et la période de temps t qui lui correspond. S'il s'agit d'un réglage à la baisse,

on augmentera le coefficient ω_t d'un petit incrément de manière à augmenter la puissance totale de référence qui correspond à ce réglage. S'il s'agit d'un réglage à la baisse, on effectuera évidemment la manoeuvre inverse.

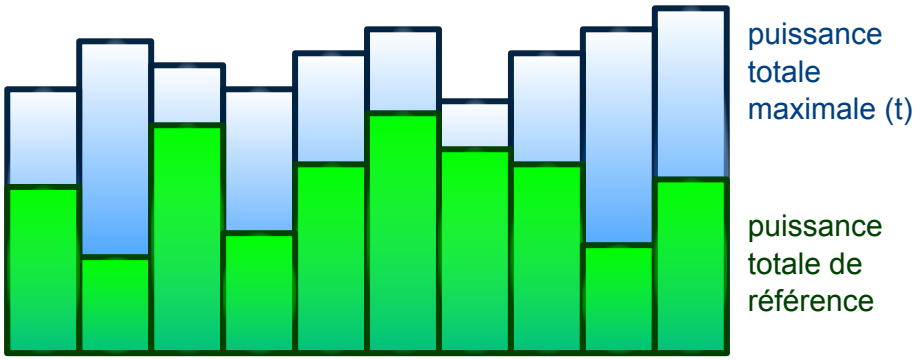

FIGURE 6.2 – Illustration de la *puissance totale de référence* et de la puissance totale maximale en fonction du temps.

Il reste à préciser la notion de **petit incrément** mentionnée précédemment. On comprend bien que plus celui-ci sera petit, meilleure sera la solution finale si on laisse à l'algorithme suffisamment de temps. Le choix de l'incrément est donc directement lié au nombre d'itérations que l'on permettra. Finalement, ce sera à l'utilisateur de décider son propre **compromis** entre qualité de la solution et temps accordé. En pratique, un pas entre 0.01 et 0.05 a été utilisé avec 1000 itérations maximum dont 100 à 200 sans améliorations de la solution.

6.5 Complexité

Le calcul de la complexité est fort proche de celui pour l'algorithme de résolution par "niveau cible".

On commence encore une fois par une vérification de la réalisabilité d'ordre $M.T$.

Vient ensuite la solution pour le dernier scénario. On considère chaque période séparément. A chaque fois, on va commencer par un tri des charges et finir par une récupération de réalisabilité. La complexité de la recherche de la solution pour le dernier scénario est donc de $T.(M.\log(M) + M.T)$.

Il ne reste plus qu'à étendre la solution aux autres scénarios et encore une fois, ceci est accompli avec une complexité de l'ordre de $(S-1).M.T^2$.

Pour conclure, on obtient donc une complexité globale de l'ordre de $M.T.\log(M) + S.M.T^2$ dans le pire des cas. Ce résultat est à multiplier par le nombre d'itérations si l'on décide de changer les coefficients de référence.

6.6 Démarrage *"à chaud"*

Et si l'on change légèrement les données d'un problème déjà résolu, doit-on recommencer tout à zéro ? Et bien non, un système de démarrage *"à chaud"* est envisageable. Entre les différentes itérations de la recherche locale, seuls les coefficients ω_t changent entre les différentes itérations. On peut, au lieu de démarrer avec tous les coefficients à 0.5, choisir de partir des coefficients déterminés précédemment. Nous définirons alors un nombre d'itérations maximum plus faible ainsi qu'un pas plus petit.

7 Algorithme d'amélioration locale

7.1 Présentation

Précédemment, nous tentions de trouver une tendance globale pour toutes les charges à atteindre et ensuite, nous appliquions cette idée générale à toutes les charges. Dans ce dernier algorithme, nous effectuerons la démarche inverse.

Cette fois, nous considérerons chaque charge individuellement et déterminerons comment les activer au mieux. Ensuite, on examinera comment celles-ci interviennent dans la solution finale.

Ici, nous commencerons à partir d'une solution initiale que nous souhaiterions la meilleure possible et effectuerons divers ajustements afin de tenter d'améliorer cette solution. La solution initiale pourra être fournie par n'importe quels autres algorithmes.

7.2 Faiblesse d'une charge à une période de temps

Cet algorithme fonctionnera en plusieurs itérations. A chacune d'entre elles, une charge sera modifiée : celle que nous jugerons la plus "**faible**" et pouvant être renforcée de la manière présentée dans la section suivante.

Déterminons ce que nous appellerons la *faiblesse d'une charge* i au temps t. Afin de trier aisément, on attribuera à chaque "*faiblesse*" une valeur. Celle-ci sera liée à l'apport en quantité de réglage associée au temps t.

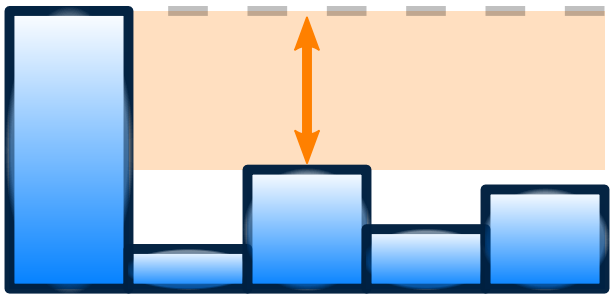

FIGURE 7.1 – Calcul de la *faiblesse d'une charge* pour un réglage à la baisse.

Pour un réglage à la baisse, on aura logiquement l'apport :

$$apport^{-}_{i,t} = \max_{\tau \in [t+1,\dots,t+N]} power^{(2t+1)}_{i,\tau} - power^{(2t+1)}_{i,t}$$

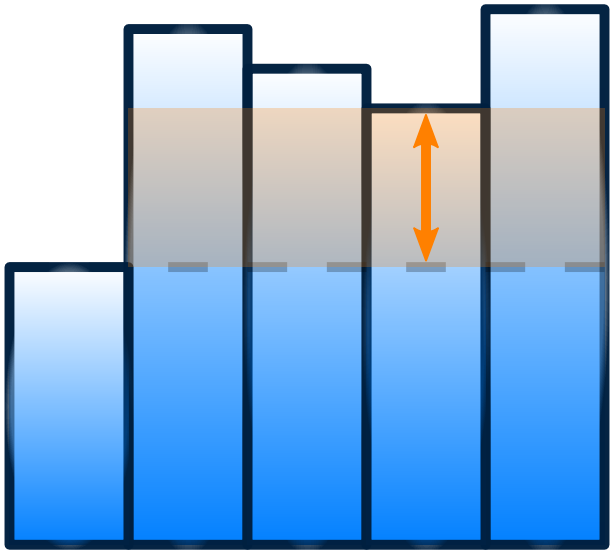

FIGURE 7.2 – Calcul de la *faiblesse d'une charge* pour un réglage à la hausse.

Pour un réglage à la hausse, la quantité associée est égale à :

$$apport^{+}_{i,t} = power^{(2t)}_{i,t} - \min_{\tau \in [t+1,\dots,t+N]} power^{(2t)}_{i,\tau}$$

Plus cette mesure sera grande, plus la charge contribuera à la solution en terme de réglage.

Il va maintenant falloir construire une valeur image de la faiblesse choisie de manière à ce que, plus celle-ci est petite, plus la charge désavantage la solution. La calcul de la faiblesse s'étant imposé est le suivant :

$$faiblesse_{i,t} = \min\left(e^{apport^{-}_{i,t}}.(Q^{-}_{va_t} - Q^{-}_{fi}) \ ; \ e^{apport^{+}_{i,t}}.(Q^{+}_{va_t} - Q^{+}_{fi})\right)$$

Comme on choisit une valeur de faiblesse pour une période de temps où deux types de réglages sont associés, il faudra faire un choix entre les deux types et par conséquent, on choisira le plus faible des deux. On y retrouve la multiplication par la différence entre la quantité de réglage disponible à la période t et la quantité fixe. Cela traduit une volonté de favoriser l'augmentation de la quantité fixe. Si on est à la période entachant une augmentation de la quantité de réglage fixe, on souhaitera une amélioration à tout prix, d'où la multiplication par 0 qui définira le 0 comme la plus petite valeur possible. Ensuite, on multipliera par l'exponentielle de l'apport afin de ramener la valeur de l'*apport* à une grandeur supérieure à zéro.

7.3 Renforcement de la charge

Établissons un système permettant de renforcer la solution en modifiant les décisions pour une seule charge i associée à une faiblesse en t. Deux cas se présentent : une faiblesse en réglage à la hausse ou une faiblesse en réglage à la baisse.

7.3.1 Renforcement du réglage à la baisse

Généralement, si l'on a un mauvais réglage à la baisse au temps t, c'est que les différences entre la puissance au temps de référence t et les puissances aux temps suivants sont faibles. En particulier, la différence la plus mauvaise sera obtenue avec le maximum des puissances survenant après t.

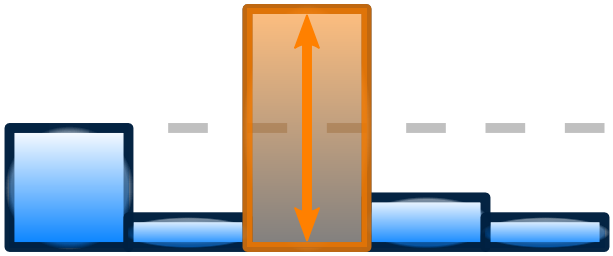

FIGURE 7.3 – Recherche du maximum de puissance après la période de référence ...

Si la solution que l'on examine a été correctement construite, les puissances après la période de référence sont les puissances minimums et nécessaires afin d'obtenir une solution réalisable. On va donc tenter de transposer cette puissance nécessaire à une période de temps précédente et ainsi obtenir une quantité de réglage à la baisse plus importante.

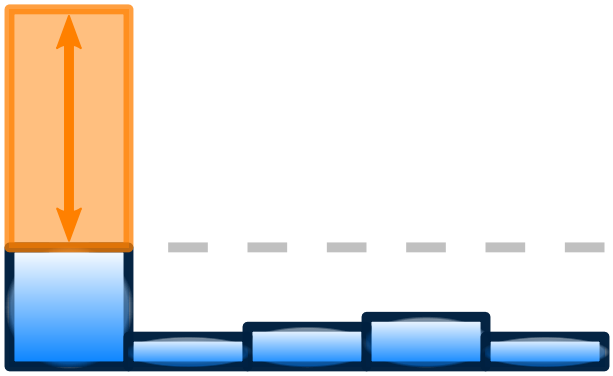

FIGURE 7.4 – ... et tentative de transposition à la période de référence.

Ainsi, on tente de gagner sur les deux tableaux. Non seulement, on tente d'augmenter la puissance de référence mais également de diminuer celles aux temps suivants. Cependant, rien ne nous assure que l'on pourra obtenir cet incrément à la période t. Si ce n'est pas le cas, on essayera alors d'obtenir cet incrément de puissance aux périodes précédentes.

7.3.2 Renforcement du réglage à la hausse

Évidemment, la solution pour un réglage à la hausse est fort similaire. Tout d'abord, nous allons tenter de diminuer la puissance à la période de référence pour tenter, comme pour le réglage à la baisse, de gagner sur les deux tableaux.

Si, malheureusement, on ne peut encore diminuer la puissance à la période de référence, on tentera de la diminuer à une période précédente. Pour cela, on va à nouveau rechercher la puissance

minimale sur les périodes de réglages et déterminer le décrément de puissance à obtenir comme étant la différence entre la puissance maximale et cette puissance minimale.

7.4 Complexité

Tout commence avec le tri des faiblesses au nombre de $M.T$ qui conduit donc à une complexité de l'ordre de $M.T.\log(M.T)$. Ensuite, on ne modifie qu'une seule charge à une période de temps. Avec l'algorithme de récupération de réalisabilité, la complexité maximale est de l'ordre de T. Finalement, on applique le résultat à tous les scénarios pour cette charge pour une complexité $S-1$.

Au total, cette méthode a une complexité de seulement $M.T.\log(M.T)+(S-1)$ à multiplier par le nombre d'itérations. On comprend pourquoi son utilisation est très légère et qu'il serait dommage de s'en priver.

8 Tests et résultats

8.1 Aperçu des différents algorithmes

Dans cette partie, nous observerons la solution obtenue par les différents algorithmes et les comparerons entre elles. Les notations "*niveau cible+*" et "*puissance cible+*" signifient que la solution a été donnée par l'algorithme correspondant avec une recherche locale, puis améliorée à l'aide de l'algorithme d'amélioration locale.

Regardons les solutions obtenues pour 4 problèmes similaires avec des nombre de charges et nombre de périodes différents. Pour ces tests, la valeur de N à été mise arbitrairement à 1. Seront présentées pour chaque problème, la somme des quantités de réglage fixe obtenues, les quantités de réglages à la baisse $Q^{-}_{va_t}$ ainsi que celles à la hausse $Q^{+}_{va_t}$.

8.1.1 100 charges et 24 périodes

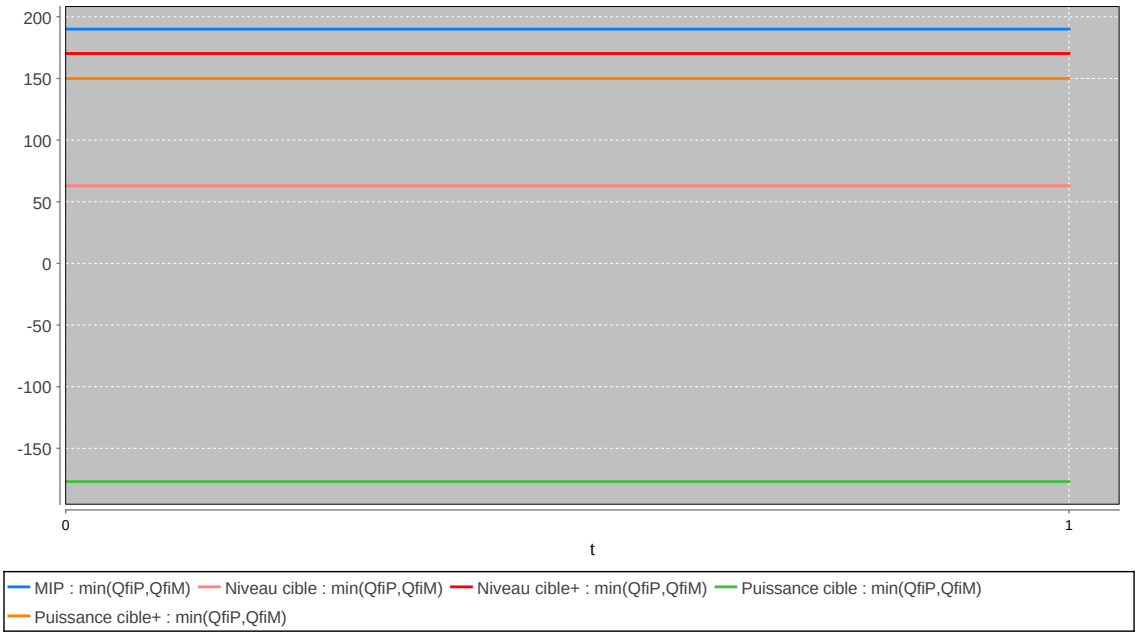

FIGURE 8.1 – Comparaison du minimum des quantités fixes obtenues sur un même problème par les différents algorithmes.

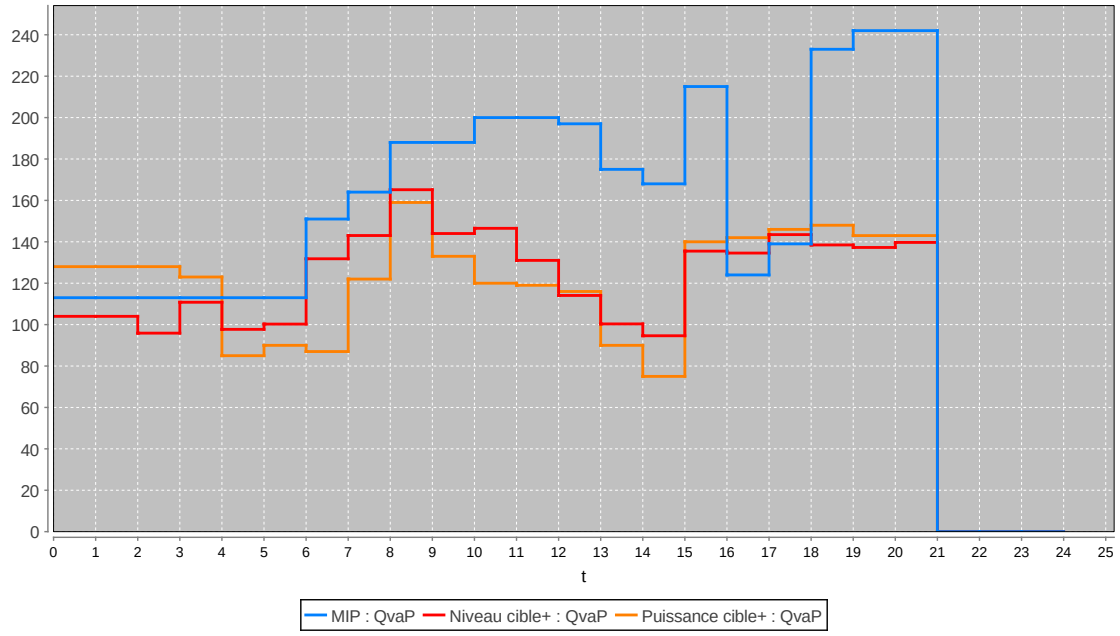

FIGURE 8.2 – Comparaison de la valeur de $Q^{-}_{va_t}$ pour les différents algorithmes.

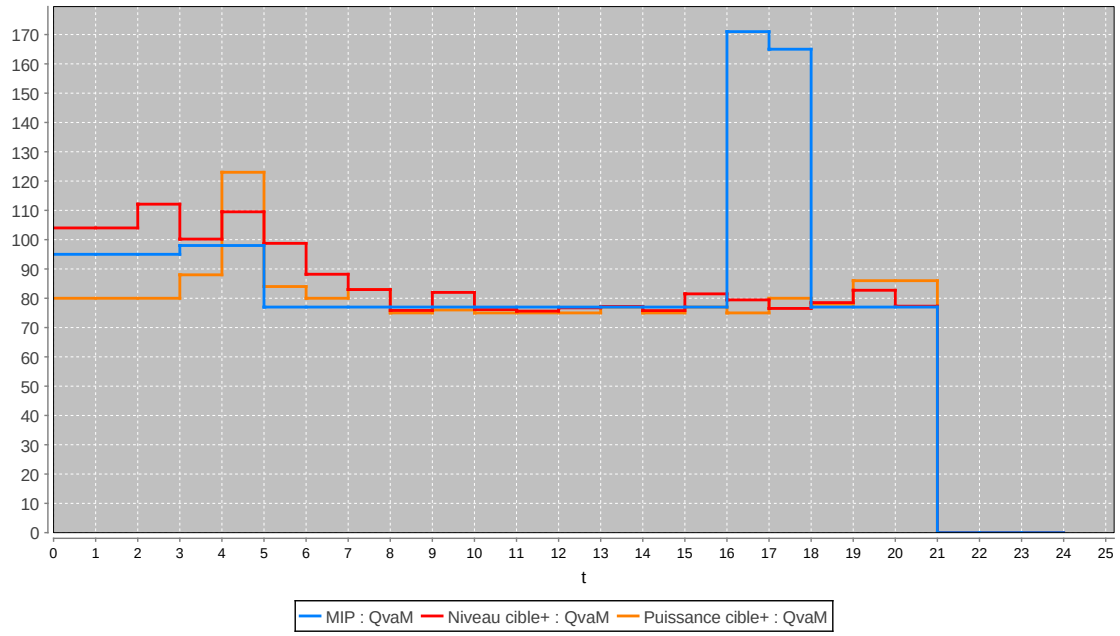

FIGURE 8.3 – Comparaison de la valeur de $Q^{+}_{va_t}$ pour les différents algorithmes.

Pour un petit problème, l'algorithme de résolution par niveau cible semble être le meilleur choix. On remarque à la figure 8.1 que l'algorithme de résolution par puissance cible sans recherche locale atteint des performances médiocres alors que celui par niveau cible s'en tire à bon compte.

8.1.2 50 charges et 48 périodes

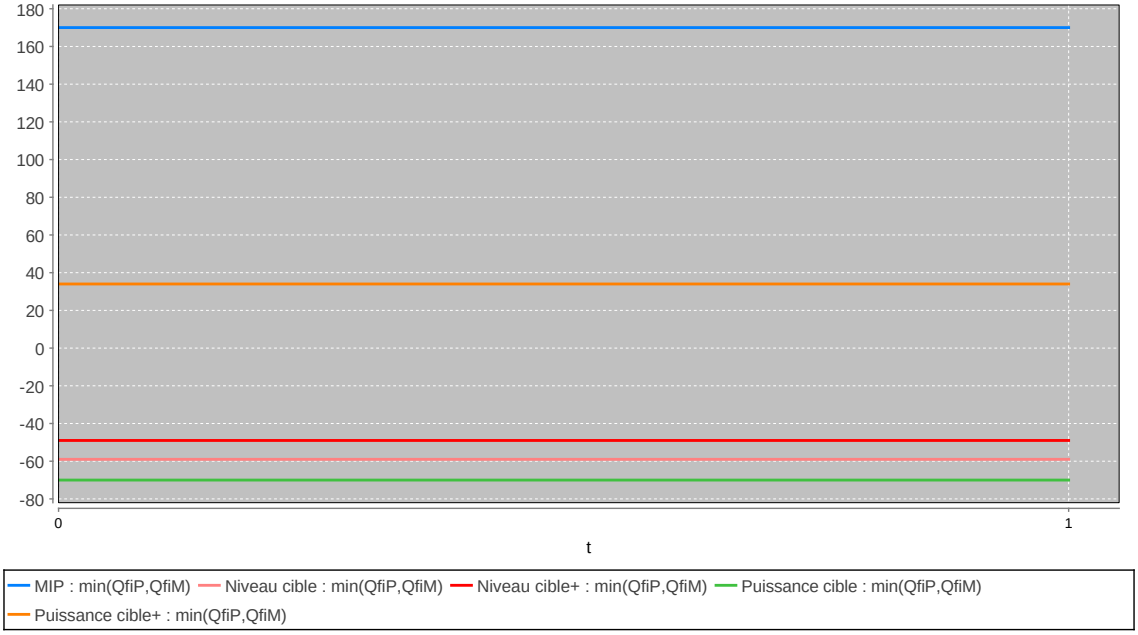

FIGURE 8.4 – Comparaison du minimum des quantités fixes obtenues sur un même problème par les différents algorithmes.

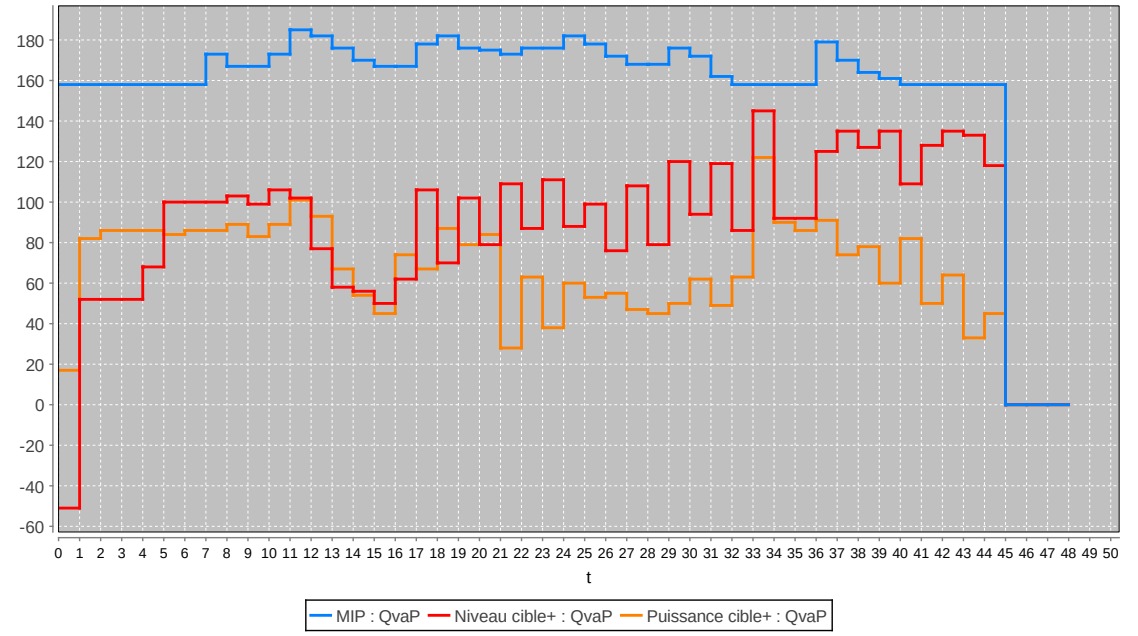

FIGURE 8.5 – Comparaison de la valeur de $Q^{-}_{va_t}$ pour les différents algorithmes.

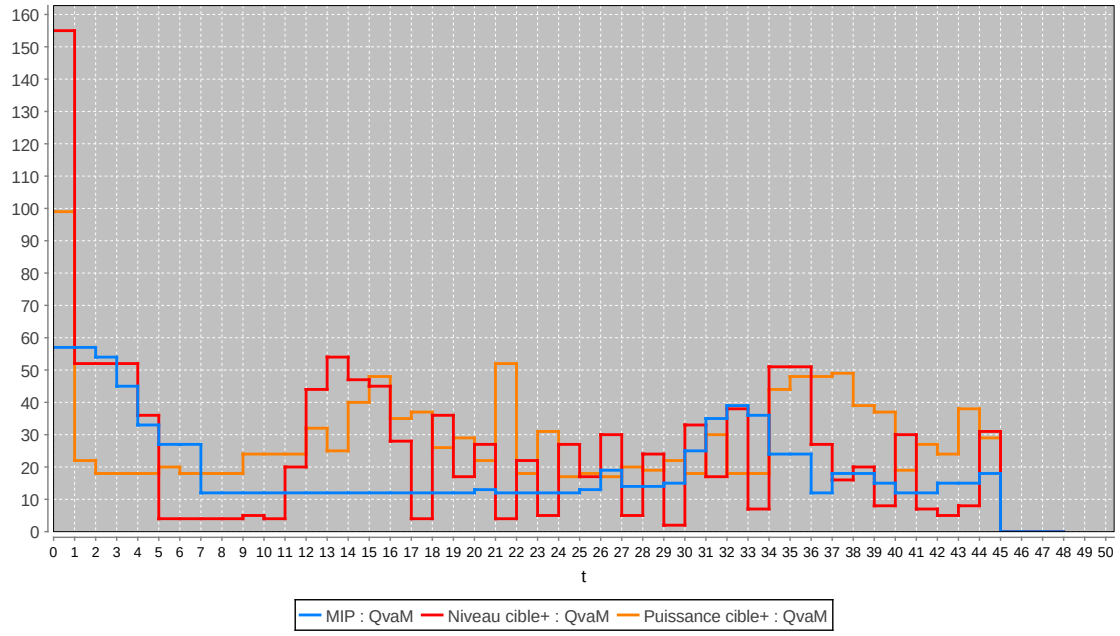

FIGURE 8.6 – Comparaison de la valeur de $Q^{+}_{va_t}$ pour les différents algorithmes.

Dès que l'on augmente un peu le nombre de périodes, les résolutions de type *puissance cible* produisent une solution nettement meilleure. L'algorithme de recherche par niveau cible nous fournit, dans ces cas-là, une solution négative et donc totalement inutilisable en pratique.

8.1.3 50 charges et 96 périodes

Cette fois, nous ne pourrons pas nous référer à la solution optimale étant donné que la taille du problème est trop conséquente pour utiliser l'algorithme du *branch and bound*.

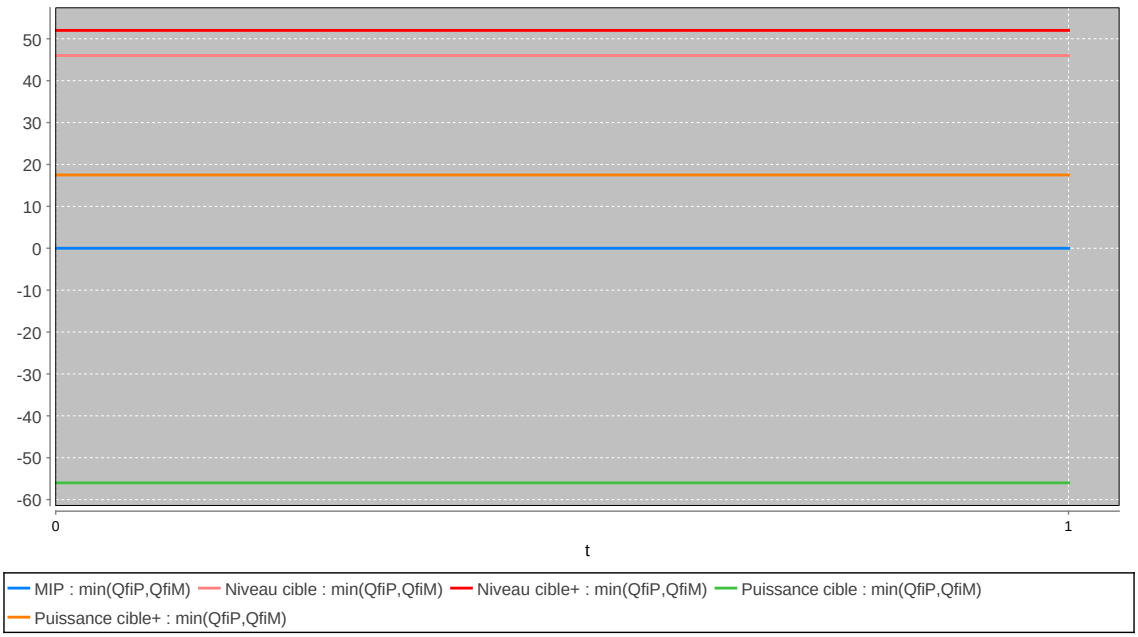

FIGURE 8.7 – Comparaison du minimum des quantités fixes obtenues sur un même problème par les différents algorithmes.

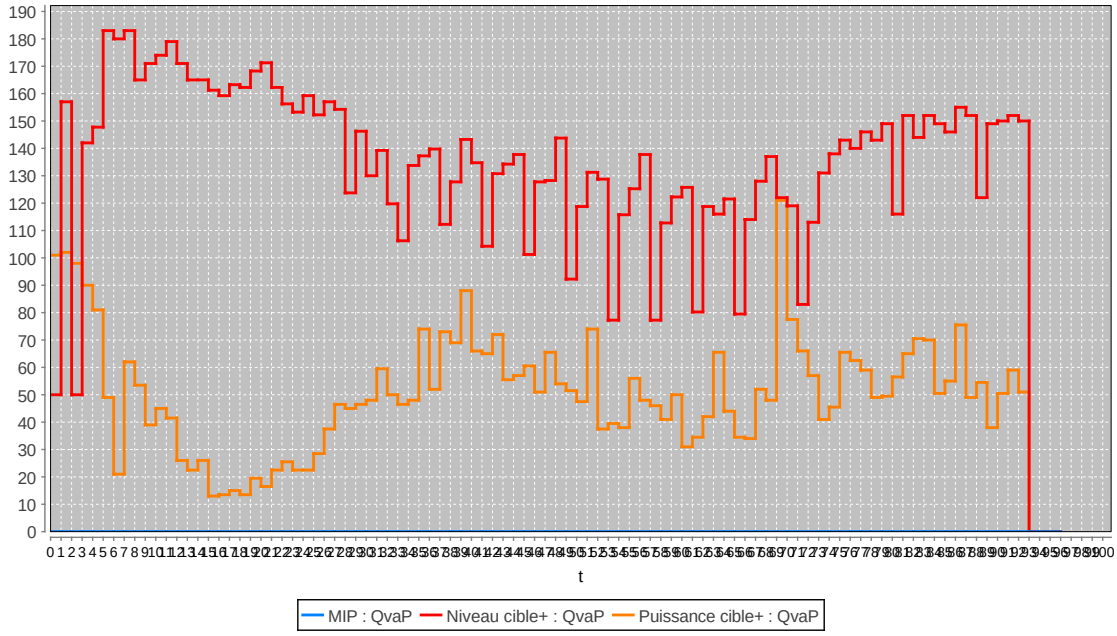

FIGURE 8.8 – Comparaison de la valeur de $Q^{-}_{va_t}$ pour les différents algorithmes.

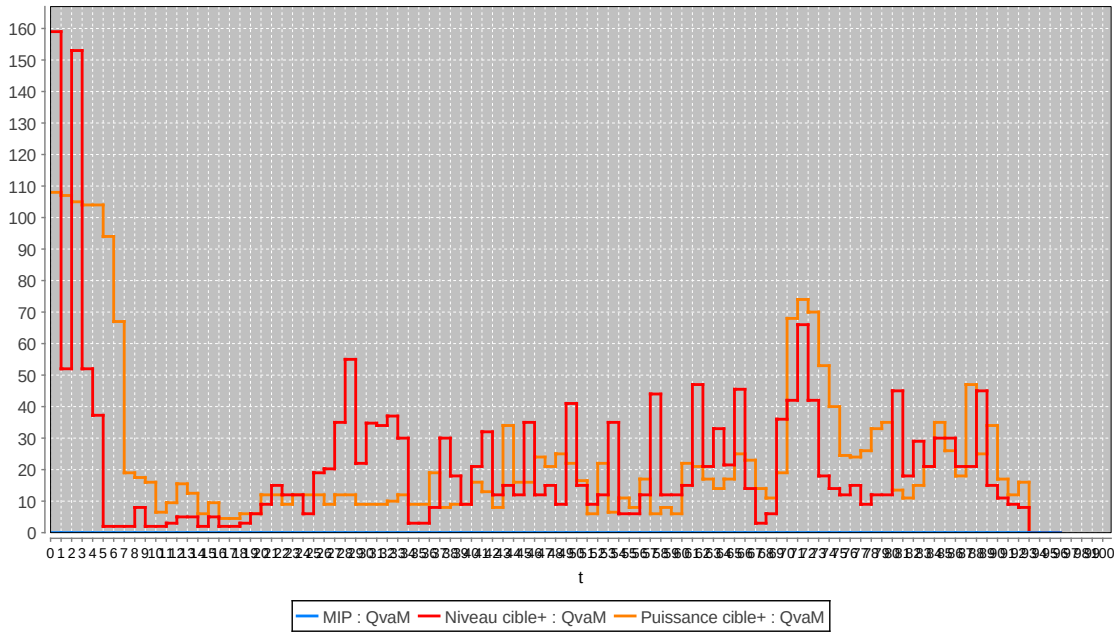

FIGURE 8.9 – Comparaison de la valeur de $Q^{+}_{va_t}$ pour les différents algorithmes.

On remarque ici que l'algorithme de résolution par niveau cible fonctionne remarquablement bien. Cela suggère, à l'utilisation et si le temps le permet, d'essayer les deux meilleures méthodes proposées et de choisir la meilleure solution des deux fournies.

8.1.4 1000 charges et 24 périodes

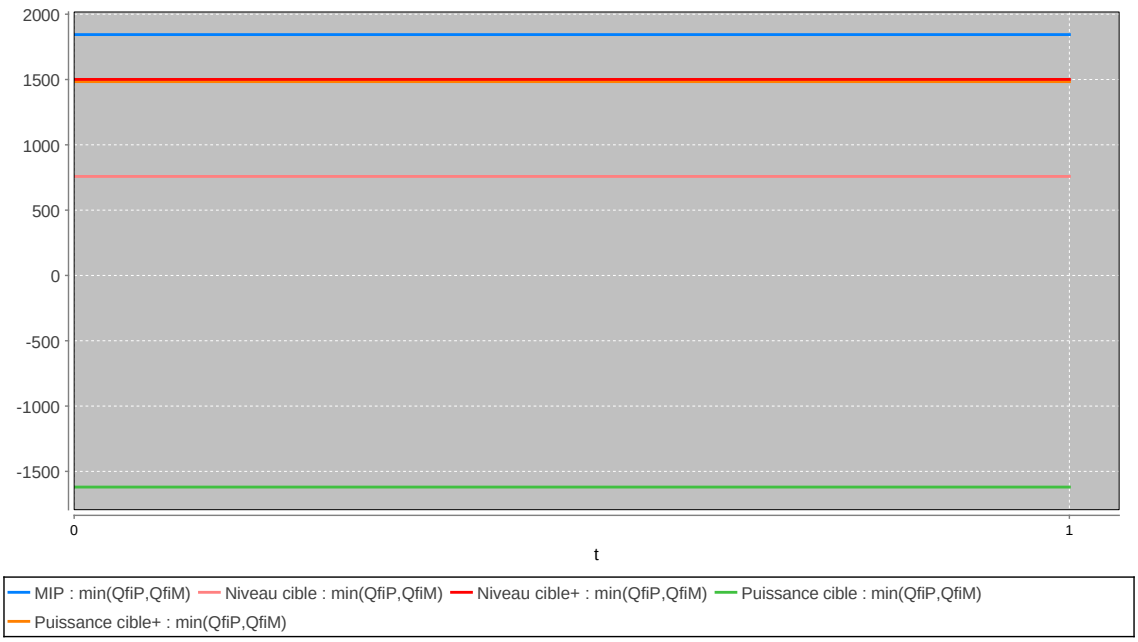

FIGURE 8.10 – Comparaison du minimum des quantités fixes obtenues sur un même problème par les différents algorithmes.

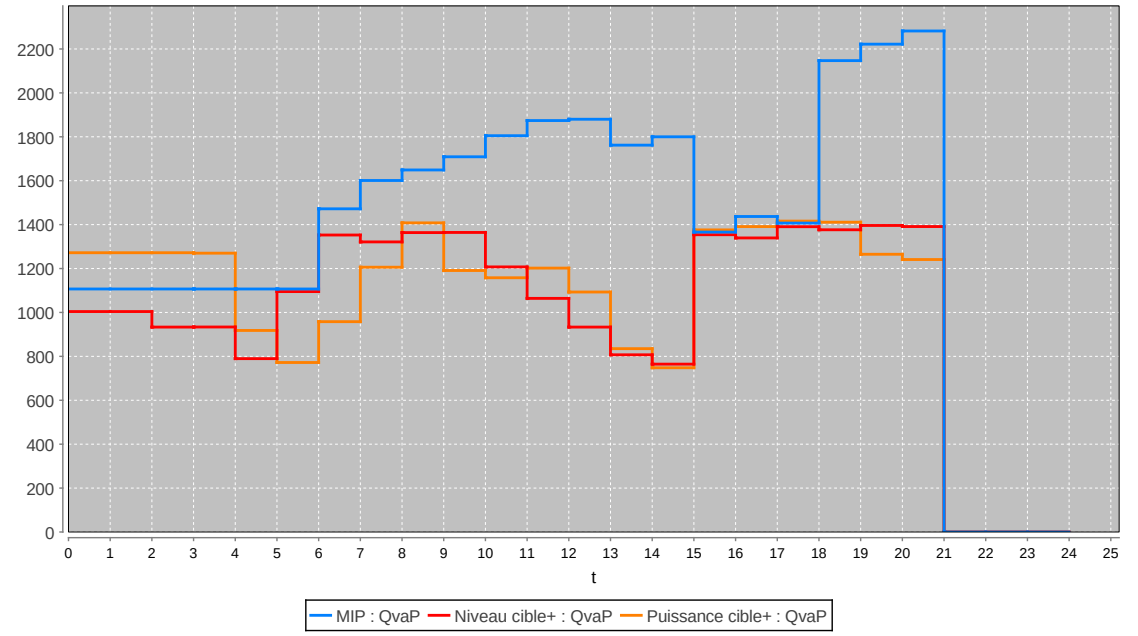

FIGURE 8.11 – Comparaison de la valeur de Q^{-}_{fi} pour les différents algorithmes.

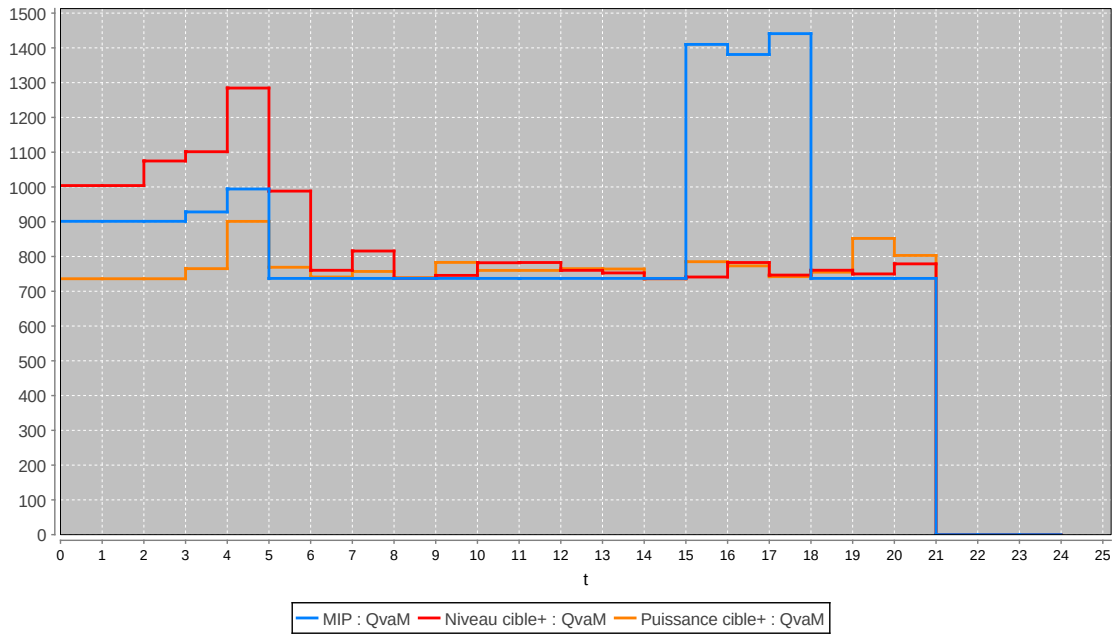

FIGURE 8.12 – Comparaison de la valeur de Q_{fi}^{+} pour les différents algorithmes.

Pour ce problème, on remarque que les meilleures versions de nos algorithmes sont au coude à coude, dépassées de peu par la solution optimale dans le réglage à la baisse. On notera la très bonne performance de l'algorithme par niveau cible sans recherche locale pour ce problème si l'on prend en compte qu'il est, dans ce cas, 500 fois plus rapide que les versions effectuant une recherche locale.

8.2 Qualité de la solution

Dans cette partie, nous allons essayer d'obtenir un aperçu de la qualité de la solution. Pour cela, un histogramme de ce qui sera appelé la *qualité de la solution* semble être une bonne idée. Nous nous intéresserons à la somme des quantités fixes obtenues via les meilleures versions de nos algorithmes et celui donné par le solveur d'optimisation en nombre entiers *CPLEX*. La qualité de la solution sera définie comme le rapport de ces deux valeurs.

8.2.1 Algorithme de résolution par puissance cible

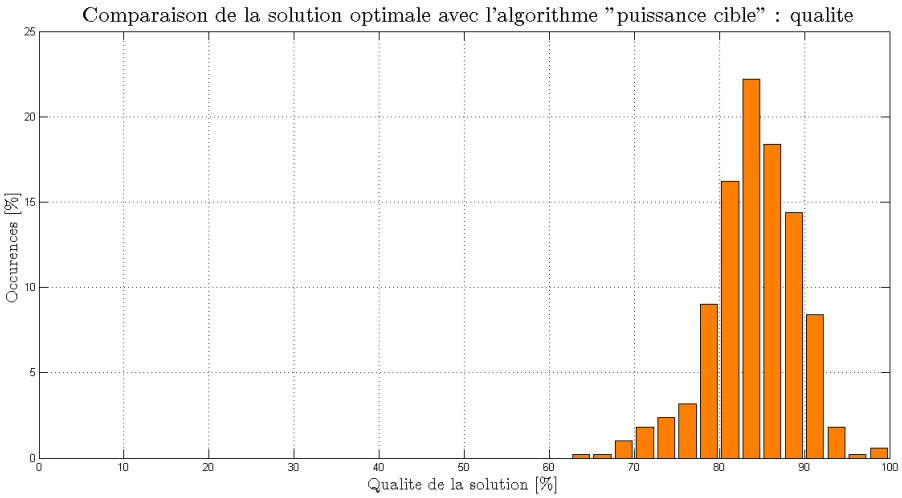

FIGURE 8.13 – Histogramme de la qualité de la solution sur 500 problèmes générés aléatoirement pour 100 charges et 24 périodes de temps.

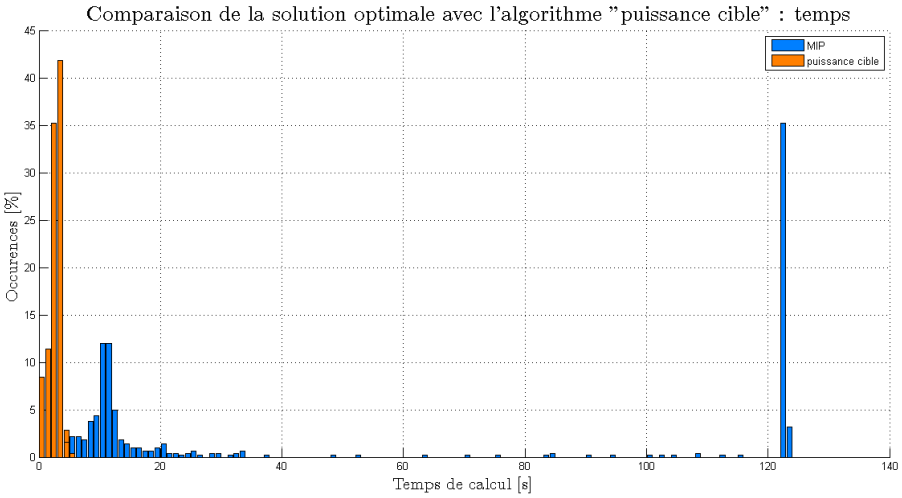

FIGURE 8.14 – Histogramme du temps nécessaire à la résolution du problème. Le temps nécessaire à la recherche de la solution optimale a été limité à 2 minutes.

On remarque à la figure 8.15 que la solution obtenue est d'une qualité généralement comprise entre 78% et 92%. Si l'on regarde, à la figure 8.16, le temps nécessaire à la résolution de ces problèmes,

on peut remarquer l'avantage conséquent de l'algorithme proposé. Notons que les tests effectués désavantagent fortement notre solution. L'implémentation effectuée n'a pas été poussée jusqu'au *multi-threading* et ne permet pas d'exploiter les 8 coeurs de la machine de tests, à l'inverse de *CPLEX*.

8.2.2 Algorithme de résolution par niveau cible

Regardons les résultats pour *l'algorithme de résolution par niveau cible* avec le même protocole de test.

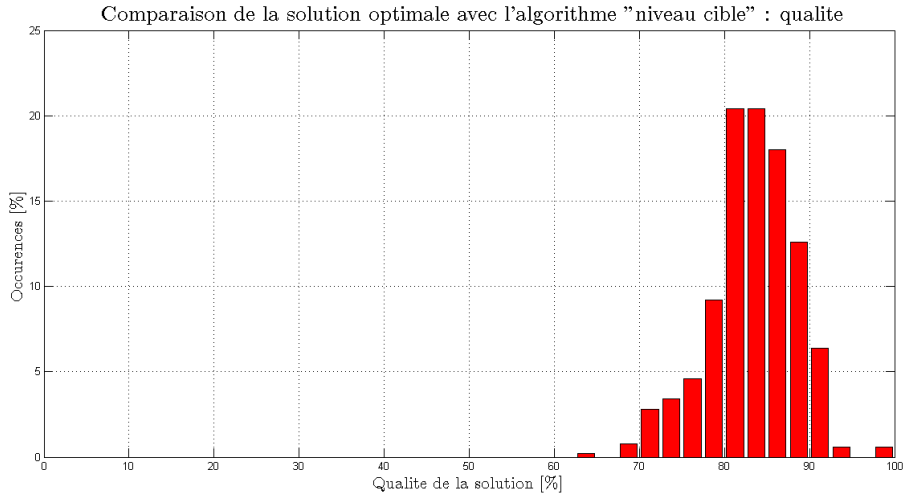

FIGURE 8.15 – Histogramme de la qualité de la solution sur 500 problèmes générés aléatoirement pour 100 charges et 24 périodes de temps.

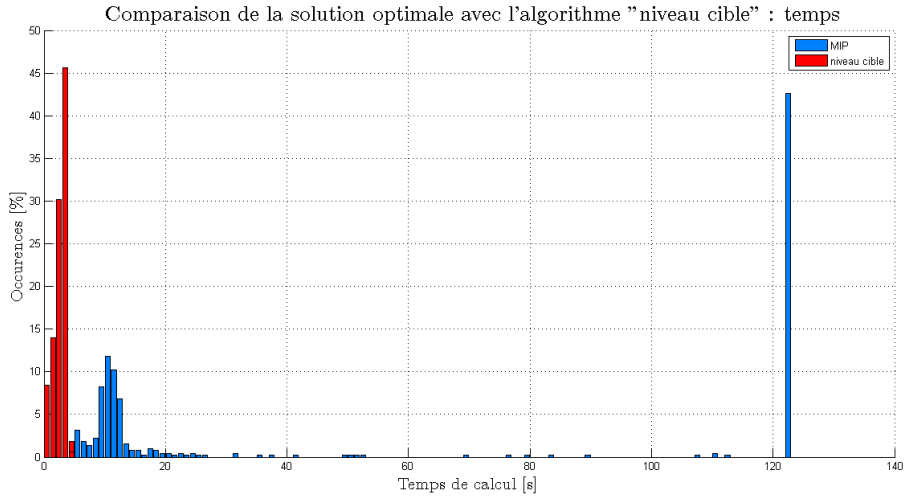

FIGURE 8.16 – Histogramme du temps nécessaire à la résolution du problème. Le temps nécessaire à la recherche de la solution optimale a été limité à 2 minutes.

La qualité de la solution obtenue est équivalente alors que le temps utilisé est à peine plus grand. Malheureusement, ces tests ont été effectués sur des petits problèmes afin d'obtenir un échantillon suffisamment important pour en déduire des statistiques fiables. D'autres tests avec des problèmes de

plus grande envergure auraient surement donnés avantage à *l'algorithme de résolution par puissance cible*.

8.3 Conclusion

Au cours du développement des algorithmes, la solution donnée par celui par puissance cible a toujours été positive alors que, dans le cas du niveau cible, certains résultats plongeaient fortement dans les négatifs, raison pour laquelle on préfèrera le premier algorithme.

Si le temps le permet, l'utilisation des deux algorithmes parait être adapté afin d'obtenir une meilleure solution. Evidemment, l'utilisation de l'algorithme du *branch and bound* reste l'option optimale si le temps ne nous est pas compté.

Notons une petite observation. Dans tous les tests précédents, l'algorithme d'amélioration locale a été employé comme *"coup de pouce"* à une solution pré-existante de qualité. On effectuait une centaine d'itérations. Dans certains cas, on peut encore l'améliorer en effectuant un nombre très important d'itérations (comme par exemple 10000) et obtenir, après un quelque temps sans changement, des améliorations conséquentes. Néanmoins, cela n'arrive pas pour tous les problèmes et ne semble donc pas être assez fiable que pour être utilisé comme tel. Cependant, comme il est possible de l'arrêter à tout moment une fois une première solution obtenue, on ne perd rien à le laisser effectuer autant d'itérations que le temps attribué à la résolution nous le permet.

9 Sélection des charges pour une quantité de réglage donnée

9.1 Explication du problème

Le but de cette partie est de développer une méthode capable, à partir d'un parc de charges accessibles, de commander ce parc afin d'approcher aux mieux une quantité de réglage déterminée, tout en minimisant la gêne des différentes charges. On voudra également connaître le nombre de périodes où cette quantité peut être atteinte.

Même si les données du problème sont presque identiques, l'objectif est différent, d'où la nécessité d'un nouvel algorithme. Ici, nous n'aurons plus la difficulté liée aux scénarios, ce qui rend le problème beaucoup plus simple à résoudre à la fois conceptuellement et au niveau de la complexité algorithmique.

Notons une petite observation mise en évidence lors de la réalisation de cet algorithme : *si l'on peut assurer une quantité de réglage de, imaginons,* $100\ MW$*, cela ne signifie pas nécessairement que nous pouvons effectuer un réglage* $5\ MW$. En effet, une quantité de réglage Q_t au temps t par rapport à un temps de référence t_{ref} est donnée par :

$$\sum_i power_{i,t_{ref}} = \sum_i power_{i,t} - Q_t$$

avec Q_t positif pour un réglage à la hausse. Prenons pour exemple que toutes les charges au temps de référence sont coupées, la puissance totale sera donc nulle. Si, pour les besoins de certaines charges, il est obligatoire de les allumer au temps t, ces charges vont alors créer une quantité minimale de réglage sous laquelle on ne pourra descendre compte tenu du passé des charges.

9.2 Algorithme de résolution

Les données exactes du problème sont identiques au problème précédent si l'on enlève la dimension des scénarios. On aura, comme données supplémentaires, la quantité de réglage à atteindre Q_{ref} et la période de référence pour la puissance. Pour un réglage à la hausse, on aura Q_{ref} supérieur à 0 et pour un réglage à la baisse, une valeur négative. Les réglages se feront aux périodes suivantes. On va donc déterminer la puissance totale de référence $P_{t_{ref}} = \sum_i power_{i,t_{ref}}$ qui servira de base pour connaitre la quantité de réglage disponible aux périodes de temps suivantes.

Ensuite, l'algorithme est composé de trois phases :

1. Activation pour un niveau optimal

2. Classement par priorité
3. Variation de l'activation pour atteindre l'objectif

9.2.1 Activation pour un niveau optimal

Nous tenterons de minimiser la *"gêne"* du client. Nous considérerons que, dans notre cas, une gêne minimale est atteinte lorsque la charge est au niveau moyen entre son niveau minimal et maximal.

Le but ici est donc de connaitre la puissance à fournir à la charge au temps t, telle que son niveau au temps $\tau = t + 1 + \delta$ soit optimal. Il faudra évidemment obtenir une puissance valide, c'est à dire comprise dans l'intervalle $0 \cup [l, u]$.

Évidemment, tout ceci semble déjà connu. C'est exactement ce qui a été effectué précédemment, et toute la difficulté résidait dans la construction d'une solution réalisable. L'*algorithme de récupération de réalisabilité* peut donc être réutilisé tel quel, ce qui justifie doublement son développement.

9.2.2 Classement par priorité

A chaque charge, nous allons associer un *coefficient de confort*. Celui-ci est construit de telle manière que plus on s'éloignera du niveau optimal, plus il sera proche de 0. Au niveau optimal, ce coefficient devra être égal à 1.

On cherche donc une fonction $\kappa(level) : [min; max] \rightarrow [0; 1]$. Après réflexion, la fonction κ s'étant imposée est la suivante :

$$\kappa(level) = 1 - \left(\frac{max + min - 2.level}{max - min} \right)^{\psi}$$

avec ψ un multiple entier positif de 2. Cette fonction est représentée à la figure 9.1 pour différentes valeurs de ψ. En pratique, on choisira ψ égal à 2 ou 4 vu les faibles variations des valeurs par rapport au gain en nombre de multiplications.

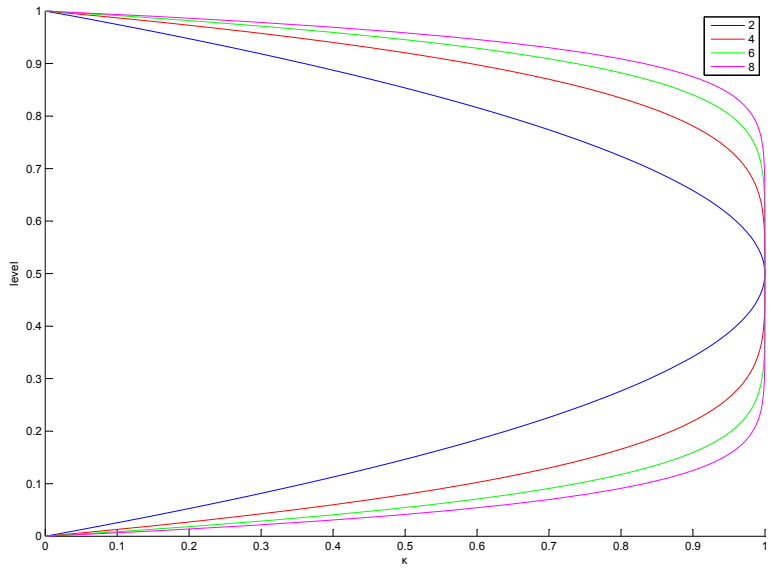

FIGURE 9.1 – Fonction $\kappa(level)$ pour différentes valeurs de ψ dans le cas d'un maximum à 1 et un minimum à 0.

9.2.3 Variation de l'activation pour atteindre l'objectif

C'est seulement à cette étape que l'on va effectuer plusieurs itérations afin d'arriver à la quantité de réglage demandée.

On définit ΔQ_t comme étant égal à la différence entre la quantité de réglage de référence et celle réellement obtenue notée Q_t. On a donc $\Delta Q_t = Q_{ref} - Q_t$ et le but évident d'arriver à annuler cette différence. Pour cela, l'algorithme va modifier la puissance de chaque charge par petits incréments fonction du *coefficient de priorité*.

Il reste à déterminer l'incrément à ajouter. On voudra évidemment, pour assurer la convergence de la méthode, un incrément proportionnel à ΔQ_t. Afin de favoriser l'activation des charges les plus proches de leur niveau optimal, il est logique de multiplier par κ et de commencer à influer sur la puissance des charges avec les valeurs de κ les plus élevées. Finalement, comme on désire répartir la puissance à obtenir sur les M charges, on va diviser notre incrément de puissance par une grandeur image du nombre de charges restant dans l'itération. En conclusion, l'incrément de puissance pour une charge i est donné par :

$$\frac{-\Delta Q_t.\kappa_i}{i+1}$$

Dans un premier temps, une version avec une variation de puissance de $\frac{-\Delta Q_t}{M}$ avait été envisagée mais s'avérait être instable à l'approche de la solution. Une itération sur l'ensemble des charges menait à une variation totale trop importante. Ce phénomène est dû à la discontinuité entre la puissance $l_{i,t}$ et 0. A l'approche de la valeur cible, ΔQ_t va prendre tantôt une valeur positive, tantôt une valeur négative. L'état des puissances des charges va donc prendre successivement les valeurs 0 et $l_{i,t}$ sans espoir de stabilisation. On ajustera donc simplement la quantité ΔQ_t en fonction de nos ajustements. Si l'on note $p_{i,t}$ la puissance d'une charge i au temps t avant ajustement et $p^*_{i,t}$ celle après ajustement **et récupération de la réalisabilité** si nécessaire. On peut dès lors déduire la valeur de ΔQ^*_t après ajustement de la charge ν :

$$\begin{aligned}
\Delta Q_t &= Q_t - Q_{ref} \\
&= \sum_{i \neq \nu} p_{i,t} + p_{\nu,t} - Q_{ref} \\
\\
\Delta Q^*_t &= Q^*_t - Q_{ref} \\
&= \sum_{i \neq \nu} p_{i,t} + p^*_{\nu,t} - Q_{ref} \\
&= \sum_{i \neq \nu} p_{i,t} + p_{\nu,t} - p_{\nu,t} + p^*_{\nu,t} - Q_{ref} \\
&= Q_t - p_{\nu,t} + p^*_{\nu,t} - Q_{ref} \\
&= \Delta Q_t - p_{\nu,t} + p^*_{\nu,t}
\end{aligned}$$

Notons quand même une difficulté. Si l'on veut augmenter la puissance d'une charge préalablement désactivée, il faudra tout d'abord l'allumer. A l'inverse, une charge dont on diminue la puissance en dessous de sa borne inférieure l devra être coupée. Évidemment, tout cela doit être effectué en préservant la faisabilité. L'algorithme correspondant est identique à celui présenté pour les solveurs précédents.

Cette partie est résumée par le pseudo-code suivant 9.1

```
dQ=(getPuissanceTotale(t)-puissanceReference)-Qref
it=0
```

```
while it < maxIteration && abs(dQ) > precision :
  for i in range(M) :
    oldPower=power[i][t];
    power[i][t]+=-dQ*kappa[i]/(i+1)
    if power[i][t] < l[i][t] :
      if dQ > 0 : # reduction en-dessous de l
        power[load][t]=0;
        on[load][t]=0;
      else : # allumage
        power[load][t]=l[load][t];
        on[load][t]=1;

    miseAjourNiveau(i,t)
    recuperationFaisabilite(i,t)
    dQ+=power[i][t]-oldPower
  ++it
```

Listing 9.1 – Récupération de la quantité de réglage demandée par itérations successives.

10 Notion d'incertitude

10.1 Introduction

Les modèles précédents supposaient une connaissance parfaite de la charge. Le niveau initial était considéré comme connu, tout comme la sortie. Malheureusement, le monde parfait n'est pas celui de la pratique. Il va donc être nécessaire d'adapter ce qui a été développé précédemment et ainsi, prendre en compte l'**incertitude** sur certaines données des charges.

10.2 Changements du modèle

10.2.1 Niveaux et sorties

Nous allons prendre en compte les incertitudes sur deux types de données :
- le niveau et par conséquent le niveau initial
- la sortie du réservoir générique

Un petit exemple pourrait être une voiture électrique dont on ne sait si celle-ci est chargée à 75% ou 80% qui nous donne donc une incertitude sur le niveau de 5%. Après un trajet avec cette voiture, on pourrait avoir une consommation, et donc une sortie, comprise entre 30% et 40%.

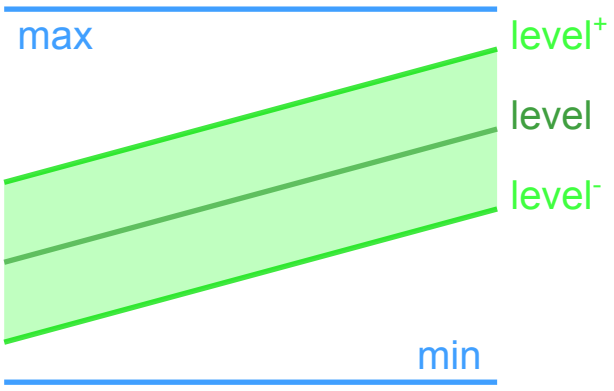

FIGURE 10.1 – Visualisation des nouvelles variables $level_{i,t}^{+,(j)}$ et $level_{i,t}^{-,(j)}$.

Il va donc falloir modifier le modèle afin de prendre en compte ces plages de valeurs possibles. Par conséquent, au lieu d'avoir une seule variable $level_{i,t}^{(j)}$, on aura les deux variables $level_{i,t}^{+,(j)}$ et $level_{i,t}^{-,(j)}$. Concernant la sortie, la variable $out_{i,t}$ va se scinder en deux variables $out_{i,t}^{+}$ et $out_{i,t}^{-}$.

A partir de probabilités

En pratique, les données des charges peuvent provenir de statistiques. Le plus souvent, les données recueillies peuvent être approximées par une loi normale $\mathcal{N}(\mu, \sigma)$ de moyenne μ et de variance σ.

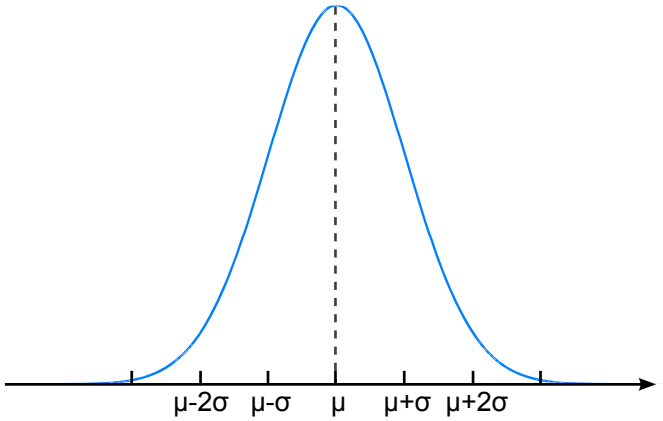

FIGURE 10.2 – Densité de probabilité d'une loi normale.

Si l'on désirait couvrir la totalité des cas possibles, il serait nécessaire de prendre un intervalle d'une taille *infinie*. On devra donc se permettre un certain degré de risque et prendre ainsi un intervalle plus petit. En prenant l'intervalle $[\mu - \sigma; \mu + \sigma]$, on couvre déjà 68% des cas possibles. Si l'on étend l'intervalle à $[\mu - 2\sigma; \mu + 2\sigma]$, notre certitude s'élève à 95%.

Évidemment, plus l'incertitude sera grande, moins il y aura de flexibilité sur la charge et moins elle contribuera à la solution. De plus, si l'incertitude est trop importante, l'algorithme pourrait ne pas trouver de solution réalisable.

10.2.2 Marges et dépassements des limites de niveau

La priorité à respecter tout au long du travail fut la robustesse de la solution par rapport aux différentes charges. On citera surtout les contraintes sur le niveau minimal et maximal devant à tout prix être respectées. Cependant, avec l'introduction de cette incertitude, peut-être serait-il bon de prévoir la possibilité de définir des *"maximum et minimum secondaires"* pouvant être atteints uniquement dans des cas critiques.

Nous allons donc introduire deux nouvelles bornes sur le niveau : max^+ et min^- définissant deux nouvelles zones critiques. Ainsi, la variable $level^+$ pourra, à certains moments, transgresser la limite donnée par le maximum tout en restant inférieure à la nouvelle borne max^+. De la même manière, $level^-$ pourra avoir une valeur inférieure au minimum tant que celle-ci reste supérieure à min^-. Nous appellerons par la suite les zones comprises entre max et max^+ ainsi que entre min et min^- les **marges**.

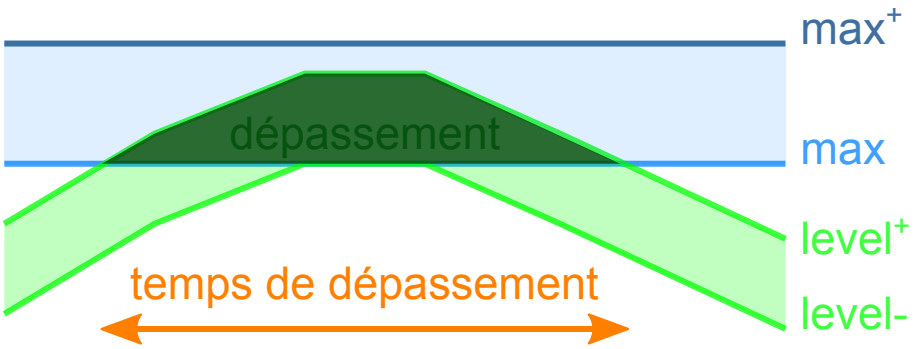

FIGURE 10.3 – Possibilité de dépassement de la variable $level^+$ au-delà du maximum dans la zone de marge comprise entre max et max^+.

Nous considérerons, dans cette modélisation, que nous accepterons un dépassement dans les marges un nombre de périodes définies suivant la période de référence d'un scénario de réglage. Ainsi, nous conservons une modélisation robuste qui ne dépasse que dans les cas spécifiques des demandes de réglage.

10.2.3 Évolution des niveaux

Dorénavant, pour une même charge, nous aurons deux niveaux à traiter. Il apparait donc que l'introduction de cette notion d'incertitude se contente de doubler la taille du problème. Si, précédemment, nous considérions le niveau initial de la charge fixé tel que $min_{i,0} = level_{i,0} = max_{i,0}$, à présent, nous commencerons avec deux niveaux initiaux donnés par : $level^-_{i,0} = min_{i,0}$ et $level^+_{i,0} = max_{i,0}$.

Il va également falloir faire évoluer la contrainte d'évolution du niveau. Pour rappel, nous avions précédemment $\forall i, t \geq \delta_i, j$:

$$\mathsf{level}^{(j)}_{i,t+1} = \mathsf{level}^{(j)}_{i,t} - \mathsf{out}_{i,t} + \left(a_{i,t-\delta_i}\mathsf{power}^{(j)}_{i,t-\delta_i} + b_{i,t-\delta_i}\mathsf{on}^{(j)}_{i,t-\delta_i}\right)$$

A présent, nous aurons les deux contraintes suivantes $\forall i, t \geq \delta_i, j$:

$$\mathsf{level}^{-,(j)}_{i,t+1} = \mathsf{level}^{-,(j)}_{i,t} - \mathsf{out}^{+}_{i,t} + \left(a_{i,t-\delta_i}\mathsf{power}^{(j)}_{i,t-\delta_i} + b_{i,t-\delta_i}\mathsf{on}^{(j)}_{i,t-\delta_i}\right)$$

$$\mathsf{level}^{+,(j)}_{i,t+1} = \mathsf{level}^{+,(j)}_{i,t} - \mathsf{out}^{-}_{i,t} + \left(a_{i,t-\delta_i}\mathsf{power}^{(j)}_{i,t-\delta_i} + b_{i,t-\delta_i}\mathsf{on}^{(j)}_{i,t-\delta_i}\right)$$

Remarquons que le niveau supérieur est associé à la sortie la plus faible, et inversement, ce qui est tout à fait logique vu le signe dans l'équation de l'évolution du niveau. Les puissances auxquelles la charge est soumise restent identiques pour les deux niveaux.

10.2.4 Adaptation des charges *tests*

Pour les voitures électriques, nous prendrons logiquement en compte une plage de valeurs de niveaux initiaux, par exemple entre 45% et 55%. La quantité de charge consommée après un trajet sera entachée d'une certaine incertitude. Si l'on ne peut aller au delà des 100% de charge, on prévoira la possibilité de descendre légèrement en dessous du niveau minimum.

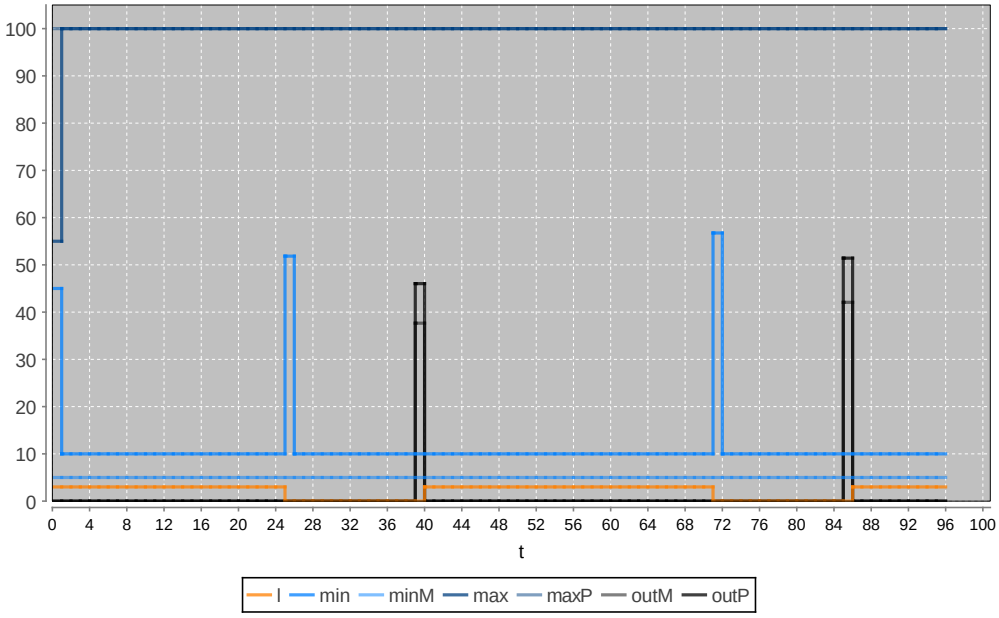

FIGURE 10.4 – Modélisation d'une voiture électrique avec une certaine incertitude.

On observe à la figure 10.4 le niveau initial entre 45% et 55%. En noir, on observe les sorties de deux valeurs différentes. On observe également que, au-delà du minimum à 10%, il existe une marge permettant de descendre, dans le pire des cas, jusqu'à 5%.

Dans le cas du chauffage, on se contentera de définir des marges de températures comme montré à la figure 10.5.

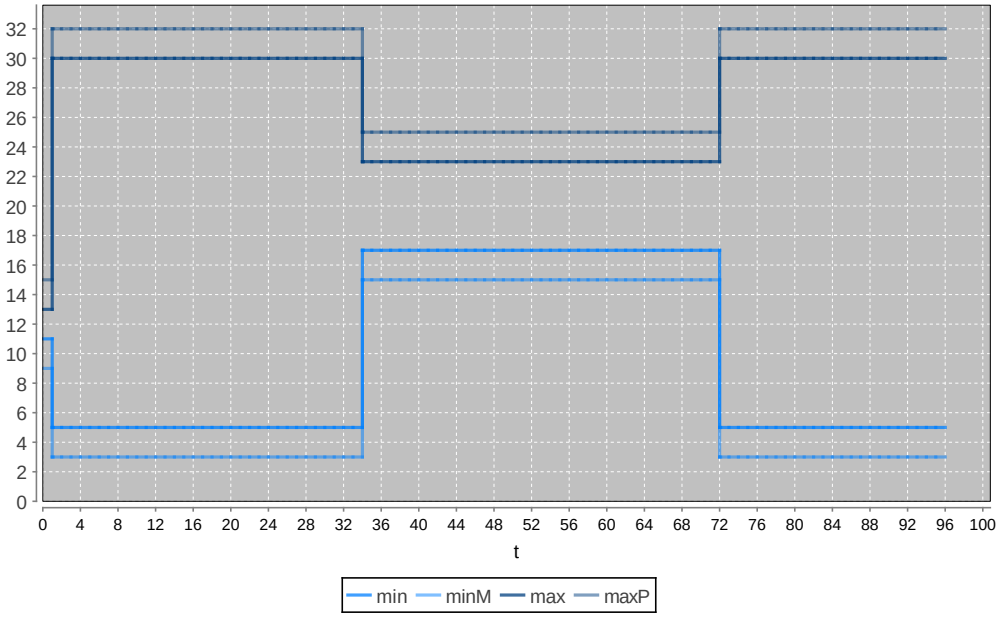

FIGURE 10.5 – Modélisation d'un chauffage avec une certaine incertitude. Pendant les heures de travail et dans des circonstances exceptionnelles, la température pourra monter jusqu'à 25 degrés ou descendre à 15 degrés.

10.3 Adaptation des algorithmes

10.3.1 Vérification de la réalisabilité

Précédemment, nous avions défini les critères permettant de déterminer rapidement si un problème est réalisable ou non. Encore une fois, il va falloir adapter les résultats précédents.

Est non-triviale uniquement l'adaptation des non-réalisabilités dues à un minimum trop élevé empêchant un maximum trop bas quelques périodes plus tard. Le cas du maximum trop haut est encore une fois symétrique. On remarque qu'il faut habilement jouer sur les deux niveaux $level^+$ et $level^-$. Considérons l'exemple donné à la figure 10.6. Le minimum en $t-2$ nous donne directement la valeur minimale de $level^-_{t-2}$ ainsi que le niveau $level^+_{t-2}$ qui lui correspond.

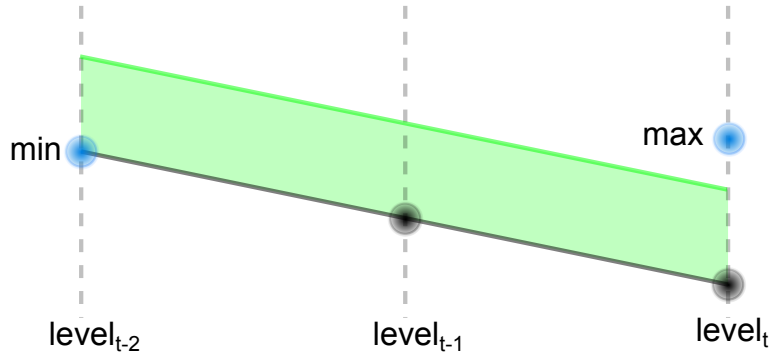

FIGURE 10.6 – Détermination des maximums possibles en considérant une certaine incertitude.

Dans le cas des maximums réalisables, il faut regarder au niveau maximal dans le cas d'une activation à la puissance la plus réduite possible. Par contre, si l'on veut utiliser le système d'amélioration des bornes, ce sera le niveau minimal qu'il faudra regarder.

Comme précédemment, il ne faut pas considérer tous les minimums aux temps inférieurs à t pour vérifier la réalisabilité d'un maximum en t.

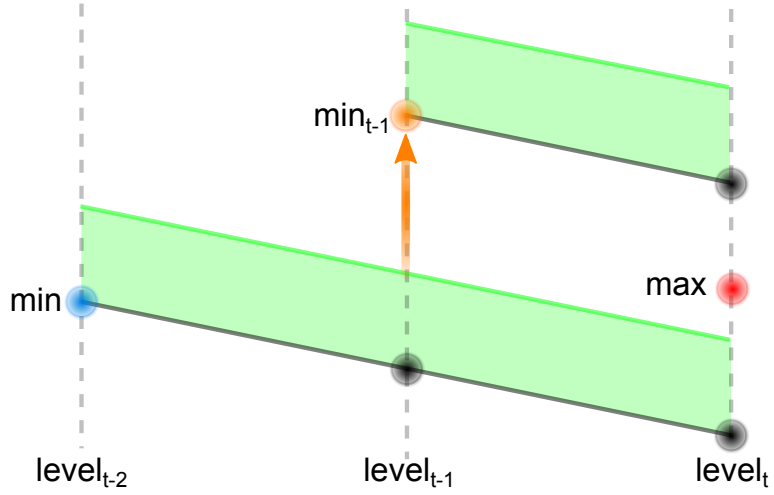

FIGURE 10.7 – Abaissement de la borne minimale des maximums grâce à un minimum en $t-1$.

Pour une meilleure compréhension, il suffit de regarder l'exemple donné à la figure 10.7. Les maximums possibles donnés à l'aide du minimum en $t-2$ se trouvent être moins contraignants que ceux déduits du minimum en $t-1$.

Malheureusement, les critères déterminés précédemment ne suffisent plus à déterminer directement si un problème est réalisable ou non. Il faudra vérifier que l'activation à la puissance minimale permet au niveau supérieur d'être en dessous du maximum. Heureusement, nous avons directement accès à cette grandeur. L'activation à la puissance minimale est ce que nous avons calculé en recherchant le maximum possible. Il suffit donc de vérifier que le niveau maximum correspondant est bien inférieur au minimum.

10.3.2 Création d'une solution réalisable

Dans les algorithmes présentés, la tâche la plus ardue était la création d'une solution réalisable. Un algorithme récursif avait été développé afin de répondre à ce besoin. Nous allons expliquer, dans cette partie, comment l'algorithme de récupération de réalisabilité a été étendu pour prendre en compte l'incertitude. Nous en profiterons pour ajouter le système des marges. L'algorithme de récupération de réalisabilité est rappelé par le diagramme à la figure 10.8

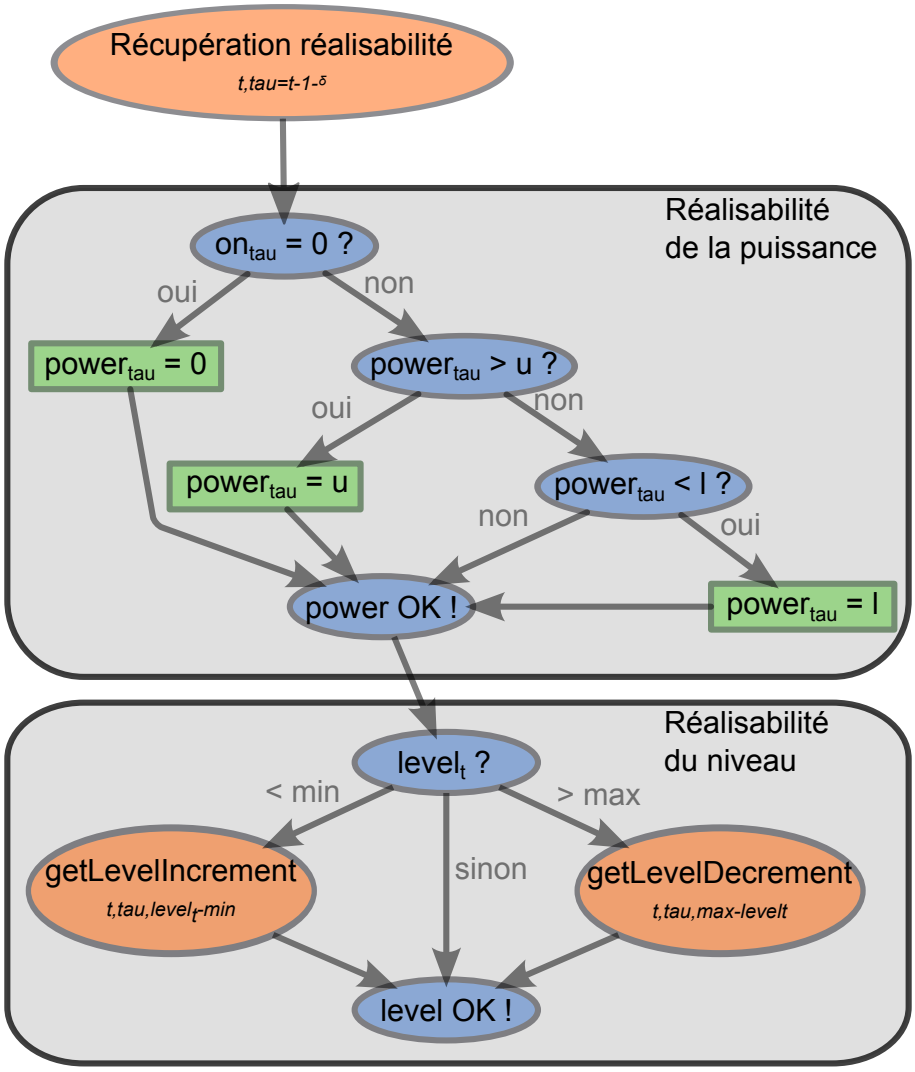

FIGURE 10.8 – Rappel du fonctionnement de l'algorithme de récupération de réalisabilité

L'adapation à l'incertitude se passe dans les fonctions *getLevelIncrement* et *getLevelDecrement* dont le but est de récupérer un incrément/décrément de niveau pour rendre la solution réalisable. Nous ne décrirons que la première, la seconde étant exactement symétrique. La prise en compte du système de marges et des deux types de niveaux est illustrée dans le diagramme à la figure 10.9

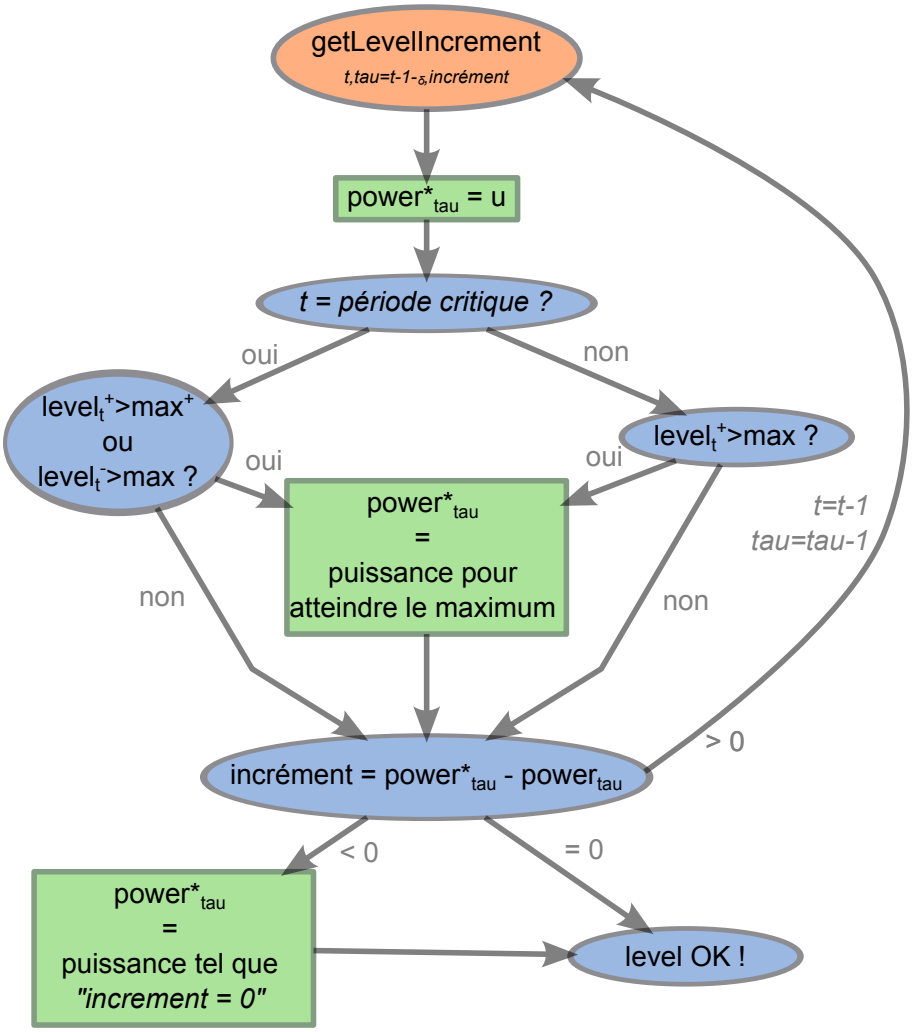

FIGURE 10.9 – Adaptation de la fonction récursive *getLevelIncrement* destinée à récupérer un incrément de niveau.

10.4 Conclusion

Heureusement, l'introduction de l'incertitude peut se faire sans changer totalement les principes sur lesquels la résolution du problème se base. Évidemment, la quantité de réglage obtenue en introduisant une incertitude ne peut être que plus faible et celle-ci doit rester modérée sous peine d'arriver à un problème non réalisable.

Après résolution, on obtient pour chaque charge une solution du type de celle donnée à la figure 10.10 pour un chauffage. On y voit les deux types de niveaux qui coexistent et sont tous deux entre les bornes minimales et maximales. On y voit également un dépassement dans la marge au moment du réglage comme permis par la modélisation.

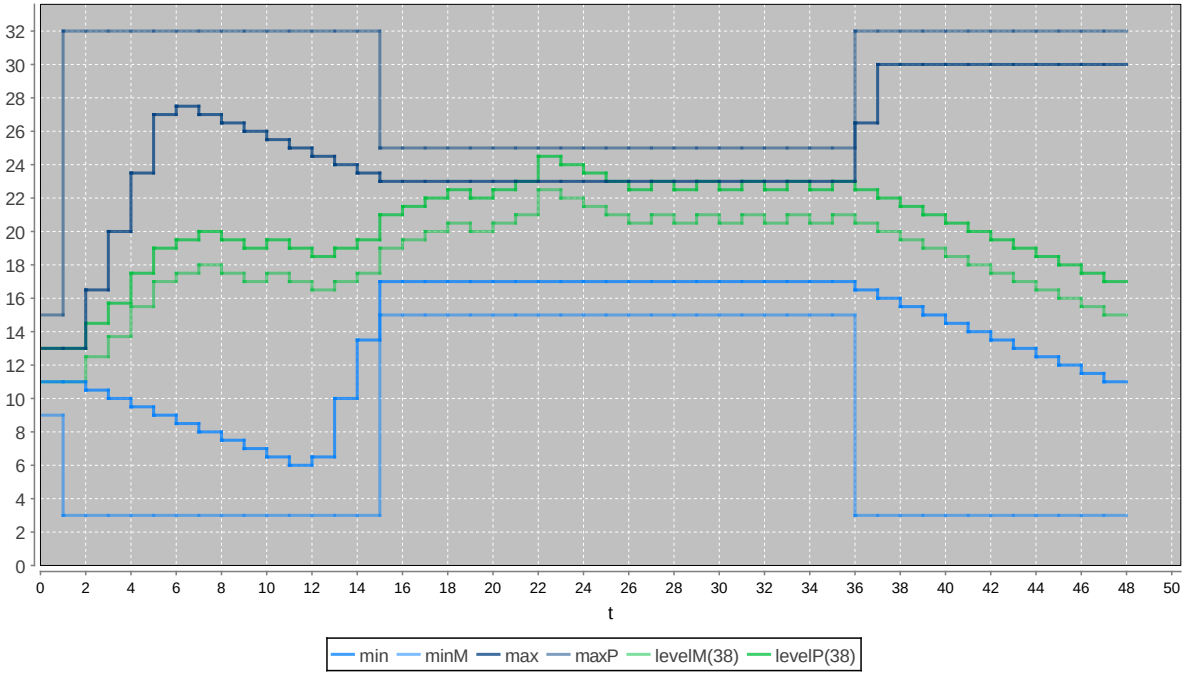

FIGURE 10.10 – Solution pour un chauffage où l'on considère une certaine incertitude.

Dans le cadre de ce mémoire, tous les outils développés précédemment ont été adaptés à la prise en compte de l'incertitude.

11 Conclusion

Tout au long de ce mémoire, nous avons apprivoisé un problème qui semblait de prime abord assez simple : obtenir une prévision des quantités de réglage disponibles en fonction du temps et ce, en commandant les puissances associées aux différentes charges. Si la modélisation était relativement aisée une fois le but du travail bien défini, la résolution en elle-même fut aussi laborieuse qu'intéressante.

Résoudre le problème à l'aide d'un solveur d'optimisation en nombre entier est aisé, une fois la modélisation terminée. La solution est d'excellente qualité voire optimale si on laisse à l'algorithme du ***brand and bound*** le temps de terminer sa besogne. Cependant, l'efficacité a un prix : tant du point de vue économique que du point de vue des performances. Pour ces deux raisons, cette méthode ne convenait pas et une alternative a été développée. Au cahier des charges, figurait la nécessité d'obtenir la solution très rapidement, c'est à dire dans un laps de temps de quinze minutes et ce, en utilisant des techniques simples afin de faciliter la maintenance.

En réalité, pas moins de 5 algorithmes alternatifs ont été développés pour finalement ne sélectionner que le meilleur d'entre eux. Bien sûr, ils ont parfois énormément en commun. Le problème reste identique et donc les solutions appliquées se ressemblent. Le point commun à toutes les méthodes de résolution développées est l'**algorithme de récupération de réalisabilité**. Celui-ci assure une solution réalisable à l'aide d'une méthode récursive.

Si l'on devait classer ces méthodes de résolution, on remarque que l'on peut distinguer celle s'occupant de modifier chaque charge individuellement de celles tentant de déterminer les décisions à prendre sur l'ensemble des charges dans leur globalité. Ces dernières méthodes de résolutions peuvent être divisées en deux catégories : celles tentant d'atteindre un **niveau de réservoir cible** et celles tentant d'obtenir une **puissance de référence cible**.

Une fois en possession d'un algorithme permettant d'obtenir une prévision efficace des quantités de réglage, une nouvelle question a été posée. **Et si l'on demande un réglage plus faible, quelle charge activer ?** A l'aide des outils conçus, la réponse a pu être obtenue aisément.

La modélisation de base était insuffisante, un critère était gênant : l'hypothèse d'une **connaissance parfaite de la charge**. Une adaptation à une certaine **incertitude** était nécessaire. Les concepts élaborés précédemment ont alors été étendus pour la prendre en compte.

Si le parc de charges évolue petit à petit pour accepter ce genre de solutions aux problèmes des réseaux électriques, sera t-il suffisamment important pour, à lui seul, contrebalancer le comportement erratique de la future production d'énergie que l'on espère totalement verte ?

Bibliographie

[1] D. Bertimas et R. Weismantel. *Optimization over Integers*. Sous la dir. de Dynamic Ideas. Belmont, 2005.

[2] CONCERE-ENOVER : Groupe de concertation Etat-Région en matière d'énergie. *BELGIQUE : Plan d'action national en matière d'énergies renouvelables*. 2010. url : http://economie.fgov.be/fr/consommateurs/Energie/Energies_renouvelables/.

[3] J.H. Eto et al. "Demand response spinning reserve demonstration". Dans : *Lawrence Berkeley National Laboratory* (2007). url : http://escholarship.org/uc/item/5m75b2gc.

[4] R. Faranda, A. Pievatolo et E. Tironi. "Load shedding : a new proposal". Dans : *Power Systems, IEEE Transactions on* 22.4 (2007), p. 2086–2093.

[5] H. Flocard et J-P. Pervès. *Sauvons le Climat*. Mar. 2012. url : http://www.sauvonsleclimat.org/etudeshtml/intermittence-et-foisonnement/35-fparticles/1161-intermittence-et-foisonnement.html.

[6] D. Kirschen et G. Strbac. *Fundamentals of power system economics*. Sous la dir. de Wiley. Jonhn Wiley & Sons, Ltd, 2004.

[7] A. H. Land et A. G. Doig. "An Automatic Method of Solving Discrete Programming Problems". Dans : *Econometrica* 28 (1960), pp. 497–520.

[8] T. Landolsi et al. "Wireless Distributed Load-Shedding Management System for Non-Emergency Cases". Dans : *International Journal of Electrical and Electronics Engineering* 4.7 (2010), p. 453–460.

[9] J.L. Lilien. *Transport et Distribution de l'Energie Electrique*. 2010. url : http://www.tdee.ulg.ac.be.

[10] Z. Luo et al. "An milp formulation for load-side demand control". Dans : *Electric machines and power systems* 26.9 (1998), p. 935–949.

[11] R.M. Maliszewski, R.D. Dunlop et G.L. Wilson. "Frequency Actuated Load Shedding and Restoration Part I - Philosophy". Dans : *Power Apparatus and Systems, IEEE Transactions on* 90 (1971), p. 1452 –1459.

[12] P. McDaniel et S. McLaughlin. "Security and privacy challenges in the smart grid". Dans : *Security & Privacy, IEEE* 7.3 (2009), p. 75–77.

[13] A.R. Metke et R.L. Ekl. "Security technology for smart grid networks". Dans : *Smart Grid, IEEE Transactions on* 1.1 (2010), p. 99–107.

[14] Clean Energy Ministerial. *Electric Vehicles Initiative (EVI)*. Avr. 2012. url : http://www.cleanenergyministerial.org/our_work/electric_vehicles/index.html.

[15] F. Rubinstein. "Using Dimmable Lighting for Regulation Capacity and Non-Spinning Reserves in the Ancillary Services Market. A Feasibility Study." Dans : *Lawrence Berkeley National Laboratory* (2011). URL : http://escholarship.org/uc/item/15z1f305.pdf.

[16] K. Samarakoon et J. Ekanayake. "Demand side primary frequency response support through smart meter control". Dans : *Universities Power Engineering Conference (UPEC), 2009 Proceedings of the 44th International*. IEEE. 2009, p. 1–5.

[17] J.A. Short, D.G. Infield et L.L. Freris. "Stabilization of grid frequency through dynamic demand control". Dans : *Power Systems, IEEE Transactions on* 22.3 (2007), p. 1284–1293.

[18] V.V. Terzija. "Adaptive underfrequency load shedding based on the magnitude of the disturbance estimation". Dans : *Power Systems, IEEE Transactions on* 21.3 (2006), p. 1260–1266.

[19] C. Warren. *Load Shedding, Load Restoration and Generator Protection Using Solid-state and Electromechanical Underfrequency Relays*. 1997. URL : http://store.gedigitalenergy.com/FAQ/Documents/F60/GET-6449.pdf.

[20] L. Wolsey. *Integer Programming*. Wiley, 1998.