Distributed optimization in large interconnected systems using ADMM

Azary Abboud

université PARIS-SACLAY

CentraleSupélec

N° d'ordre : 2016SACLC004

THÈSE DE DOCTORAT DE L'UNIVERSITÉ PARIS-SACLAY PREPARÉE À CENTRALESUPÉLEC

École doctorale N°580

"Sciences et Technologies de l'Information et de la Communication"

Spécialité : Réseaux, Information et Communications

Par

Mme Azary Abboud

Optimisation Distribuée dans les Grands Systèmes Interconnectés avec ADMM

(Distributed Optimization in Large Interconnected Systems using ADMM)

Thèse présentée et soutenue à Gif-sur-Yvette le 12 Janvier 2016:

Composition du jury :

M. Samson LASAULCE,	CentraleSupélec	Examinateur, Président du Jury
M. Damien ERNST,	Université de Liège	Rapporteur
M. Pascal BIANCHI,	Télécom ParisTech	Rapporteur
M. Walid SAAD,	Virginia Tech	Examinateur
M. Jérémie JAKUBOWICZ,	Télécom SudParis	Examinateur
Mme Houria SIGUERDIDJANE,	CentraleSupélec	Directrice de Thèse
M. Mérouane DEBBAH,	CentraleSupélec	Encadrant
M. Romain COUILLET,	CentraleSupélec	Encadrant

université PARIS-SACLAY

ÉCOLE DOCTORALE
Sciences et technologies de l'information et de la communication (STIC)

Titre : Optimisation Distribuée dans les Systèmes Larges Interconnectés

Mots clés : Optimisation distribuée, Smart Grid, DC-OPF, ADMM, Caching, Réseaux cellulaires, 5G

Résumé : Cette thèse porte sur la construction des algorithmes distribués pour l'optimisation de la production et du partage de ressources au sein d'un réseau de large dimension. Notamment, on se concentre sur les réseaux lectriques et les réseaux cellulaires 5G.
On considère dans le cas des réseaux électriques le problème OPF (Optimal Power Flow) dans lequel on vise à faire la gestion et l'optimisation de la production de l'énergie électrique d'une manière distribuée. On se concentre sur une version linéarisée du problème, la DC-OPF (Direct-Current Optimal Power Flow). Comme le problème d'optimisation est convexe dans ce cas, on vise à minimiser le coût de production de l'énergie tout en respectant les limites des lignes de transmission et les contraintes caractéristiques du système.
Dans le cas des réseaux cellulaires, on formule un problème de Caching. On a pour but de réduire l'utilisation du backhaul liant les stations de base et le contrôleur du réseau. Les stations de base sont équipées d'une capacité de stockage limitée. Ils visent à trouver d'une manière optimale les fichiers à stocker dans le but de réduire une certaine fonction de coût sur l'utilisation du backhaul et sur le partage des fichiers avec les autres stations de base.
L'approche adoptée dans cette thèse consiste à appliquer l'ADMM (Alternating Direction Method of Multipliers), une méthode d'optimisation de manière itérative, à un problème d'optimisation que l'on a préalablement reformulé de faon adéquate. Ce problème permet à la fois de décrire le DC-OPF et le problème de Caching.
On démontre la convergence de cette méthode quand elle est appliquée noeud par noeud d'une manière totalement distribuée. Ainsi que dans le cas où le réseau est divisé en plusieurs zones. Ces zones peuvent se chevaucher mais aussi elles peuvent être séparées ou indépendantes. De plus, dans le contexte d'un réseau à zones, on démontre que l'application de l'ADMM d'une manière aléatoire par une seule zone converge aussi vers la solution optimale du problème.

université PARIS-SACLAY

ÉCOLE DOCTORALE

Sciences et technologies de l'information et de la communication (STIC)

Title: Distributed Optimization in Large Interconnected Systems

Keywords: Distributed Optimization, Smart Grid, DC-OPF, ADMM, Caching, Cellular Networks, 5G

Abstract: This thesis focuses on the construction of distributed algorithms for optimizing resource production in a large interconnected system. In particular, it focuses on power grid and 5G cellular networks.
In the case of power grid networks, we consider the OPF (Optimal Power Flow) problem in which one seeks to manage and optimize the production of electrical energy in a distributed manner. We focus on a linearized version of the problem, the DC-OPF (Direct-Current Optimal Power Flow) problem. This optimization problem is convex; the aim is to minimize the cost of energy generation while respecting the limits of the transmission line and the power flow constraints.
In the case of 5G cellular networks, we formulate a caching problem. We aim to offload the backhaul link usage connecting the small bases stations (SBSs) to the central scheduler (CS). The SBSs are equipped with a limited storage capacity. We seek to find the optimal way to store files so as to reduce the cost on the use of backhaul and sharing files with other SBSs.
The approach adopted in this thesis is to apply the ADMM (Alternating Direction Method of Multipliers), an optimization method that is applied iteratively, to an optimization problem that we adequately formulated previously. This problem can both describe the DC-OPF problem and the Caching problem. We prove the convergence of the method when applied node by node in a fully distributed manner. Additionally, we prove its convergence in the case where the network is divided into multiple areas or nations that may or may not overlap. Furthermore, in the context of a network with multiple areas, we show that the application of ADMM in a random manner by a single randomly chosen area also converges to the optimal solution of the problem.

Acknowledgment

I wish to express my gratitude to a number of people that helped me throughout my three-year journey to achieve my PhD degree.

I am truly grateful to my supervisors, Prof. Mérouane Debbah, Prof. Houria Siguerdidjane and Prof. Romain Couillet, for all their help and guidance throughout my PhD studies and dissertation.

I would like to thank the jury members, Prof. Samson Lasaulse, Prof. Damien Ernst, Prof. Pascal Bianchi, Dr. Walid Saad and Dr. Jérémie Jackubowicz, for reviewing my dissertation and discussing it thoroughly.

I also would like to thank all my colleagues at CentraleSupélec, where I passed three years with daily interaction with them.

I would like to thank Prof. Tony Q.S. Quek from SUTD for all the open discussions and for exchanging ideas during my visit to SUTD.

My deepest thanks goes to my parents and family, to Hassan. I also would like to thank all my friends outside CentraleSupélec. Many thanks to all my friends and colleagues at CentraleSupélec especially Eliza, Stefan, Maria, Loig, Kenza, Matha, Abla, Emil, Karim, Ejder for all their support.

Contents

Acronyms

A	
AC	Alternating-Current
AC-OPF	Alternating-Current Optimal Power Flow
ADMM	Alternating Direction Method of Multipliers
APP	Auxiliary problem principle
B	
BS	Base Station
C	
CS	central scheduler
CRP	Chinese restaurant process
CF	collaborative filtering
CDN	content delivery network
D	
D2D	device-to-device
DC	Direct-Current
DC-OPF	Direct-Current Optimal Power Flow
DG	Distributed Generation
DPS	Distributed Production-Sharing
DR	Douglas-Rachford
E	

EMS	Energy Management System
ED	Economic Dispatch
H	
HetNet	Heterogeneous Network
HVAC	High-Voltage Alternating-Current
HVDC	High-Voltage Direct-current
I	
ISO	Independent System Operators
ICN	information-centric network
ICIC	inter-cell interference coordination
IPMs	Interior Point methods
K	
KKT	Karush-Kuhn-Tucker
KCL	Kirchoff's Current Law
L	
LP	Linear programming
LTE	long term evolution
N	
NMSD	Normalized Mean Squared Deviation
O	
OPF	Optimal Power Flow
OFDMA	orthogonal frequency-division multiple access
P	
PPP	Poisson point process
PDF	probability distribution function
PGFL	probability generating functional
PPA	Proximal Point Algorithm

Q

QP	Quadratic Programming
QoE	quality-of-experience
QoS	quality-of-service

R

RAN	radio access network

S

SC-OPF	Security constrained Optimal Power Flow
SDP	Semidefinite programming
SQP	Sequential Quadratic Programming
SLP	Sequential Linear Programming
SCADA	Supervisory Control and Data Acquisition system
SINR	signal-to-interference-plus-noise ratio
SVD	singular value decomposition
SBSs	small base stations
SCN	small cell network

T

TDD	time-division duplexing

U

UT	user terminal

List of Figures

List of Tables

Chapter 1

Résumé

Contexte et Motivation

Cette thèse se concentre sur les problèmes d'optimisation dans les systèmes larges interconnectés. Les problèmes étudiés sont généralement de grande taille et nécessitent des calculs encombrants. Normalement les problèmes à considérer possèdent certaines caractéristiques qui peuvent être exploitées par une modélisation appropriée.

Résoudre ces problèmes tout en tenant compte d'un centre de fusion central c'est-à-dire, d'une manière centralisée, est pratiquement intraitable et parfois non faisable ni fiable. Ainsi, une technique d'optimisation centralisée est prohibitive dans le cas des grands réseaux spécifiquement quand les agents sont physiquement éloignés et ont des problèmes de confidentialité.

Les méthodes d'optimisation distribuées apparaissent comme une solution attrayante dans de ce cas. Selon le problème étudié et la zone d'application, plusieurs approches peuvent être utilisées pour atteindre la cible souhaitée. Plus précisément, nous nous concentrons dans cette thèse sur deux domaines d'aplication différents. Le premier domaine d'intérêt est le réseau électrique intelligent ou Smart Grid. Dans cette partie, nous nous focalisons sur le problème des flux de puissance optimale à courant continu (*Direct-Current optimal power flow, DC-OPF*). Ceci est un problème fondamental qui est utilisé par de nombreusx opérateurs des systèmes électriques. Nous cherchons à trouver, d'une manière distribuée, la production d'énergie optimale en utilisant le minimum possible en terme de ressources. Cette solution devrait minimiser le coût de la production globale de l'ensemble du système interconnecté tout en obéissant aux caractéristiques de ce réseau. Principalement, on décrit un algorithme, "*Alternating direction Method of Multipliers* (ADMM) qui est approprié pour les réseaux statiques et dynamiques.

Cet algorithme est synchrone par défaut. Il exige une coordination entre les différents processeurs composants le système. Un inconvénient qui peut limiter les performances du système quand il est réuni avec les capacités hétérogènes de chaque processeur. Nous prouvons dans cette thèse que cet algorithme peut-être applicable de manière asynchrone. Ainsi, nous fournissons une méthode d'optimisation distribuée asynchrone où les processeurs agissent de manière indépendante.

Cette méthode a le pouvoir et lasouplesse d'être projeté dans d'autre domaines d'application juste en changeant les paramètre suivant les caractéristique du système considéré.

On se concentre sur un autre domaine d'application, celui des réseaux cellulaires 5G. Nous cherchons à trouver d'une manière distribuée le meilleur vecteur de stockage (*caching vector*) pour chaque station de base, tout en minimisant l'utilisant des liens backhaul entre ces stations et le centre de fusion "*Central Scheduler*" (CS). Nous modélisons le problème de stockage que nous visons également à résoudre par une approche distribuée en utilisant la même méthode ADMM.

Opération économique des réseaux électriques

L'énergie électrique a été découvert il ya plus de cent ans, depuis lors elle a joué un rôle majeur dans l'amélioration de notre mode de vie. Plusieurs applications et technologies ont vu le jour, de plus en plus des services sont offertess aux consommateurs que aident à garder le fonctionnement de notre société [1, 2].

La construction d'un système interconnecté comprenant des grandes stations centrales génératrices d'électricité, des lignes de transmission Haute-tension à courant alternatif (HVAC) et à courant continu (HVDC), et des lignes de distribution de basse tension AC et DC à travers les pays a donné lieu à la création d'une organisation internationale du réseau électrique.

En Europe par exemple, le réseau européen des opérateurs du système de transmission, comprend des opérateurs de 34 pays. La plupart de ces systèmes sont connectés en utilisant des technologies HVDC qui présentent une plus grande efficacité et permettent la transmission entre les systèmes AC non-synchronisés avec des pertes minimales sur les longues distances [3].

Développer et gérer à la fois les systèmes de transmission et de distribution sont des facteurs clés pour le succès dans les sociétés modernes [1,2] et jouent un rôle majeur dans la capacité du réseau européen à devenir un marché unique de l'énergie [3]. Avec pour objectif d'atteindre un réseau intelligent "Smart Grid", plusieurs améliorations ont été apportées au réseau électrique,

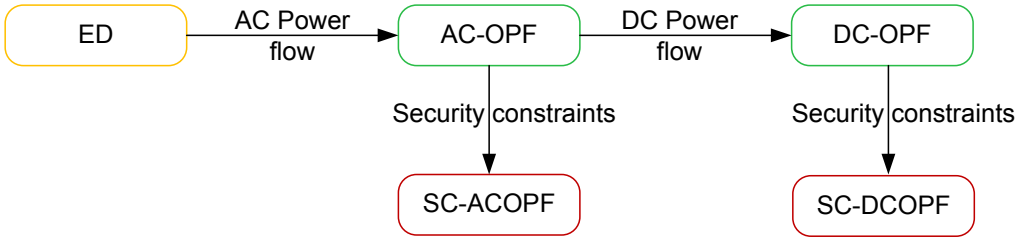

Figure 1.1: Opération économique du réseau électrique

et beaucoup d'autres sont à l'étude ou encore des perspectives.

Dans le but d'avoir un réseau intelligent, le système de service d'énergie électrique a été soumis à diverses améliorations liées à la surveillance, aux communications informatiques et à la sécurité. Ces améliorations doivent leur existence au système de gestion de l'énergie (EMS) et le système de supervision, de contrôle et d'acquisition de données (SCADA). l'EMS est utilisé par les opérateurs du réseaux afin de surveiller, de contrôler et d'optimiser la performance du système production et/ou de transmission dans une certaine zone. Le Système SCADA est un système de contrôle d'automatisation industrielle qui fournit des fonctions de surveillance et de contrôle des composants distants du système électrique.

Les mesures sur le terrain sont envoyés via le système SCADA vers le centre de contrôle. Le flux de données est comme suit, des capteurs mesurent les magnitudes et les angles de tension et les injections de puissance au niveau des noeuds . Un estimateur d'état convertit les lectures redondantes et les autres informations disponibles en une estimation de l'état réel du système tout en filtrant les bruit de mesure [4–6]. Ces estimations de l'état du réseau sont ensuite utilisées pour la surveillance et le fonctionnement réseau.

L'EMS vise à fournir un approvisionnement fiable pour les utilisateurs finales à un coût économique tout en préservant la sécurité et la stabilité du système. à cette fin, il utilise des outils tels que Economic Dispatch (ED) et Optimal Power Flow (OPF) [1, 4–8].

L'ED planifie la sortie des installations de production d'électricité de façon à répondre à la charge du système au plus bas coût d'opérations tout en respectant les contraintes opérationnelles et de tansmission [9, 10]. Il détermine la puissance générée la plus économique qui sert à servir une certaine charge. L'ED est formulé comme un problème d'optimisation convexe avec une fonction résumant les coûts individuels de chaque générateur et des contraints qui representent la réunion de la demande et de la charge du réseau.

Notre objectif ici est le problème OPF qui est très important dans la plan-

ification efficace de la production d'électricité. Les modèles de l'OPF sont essentiellement classés en i) Alternating-Current Optimal Power Flow (AC-OPF), ii) Direct-Current Optimal Power Flow (DC-OPF), et iii) Security-Constrained Optimal Power Flow (SC-OPF). Le passage d'un modèle à l'autre est représenté dans Fig.1.1.

AC-OPF, ou tout simplement OPF, intègre les contraintes AC du réseau de transmission à l'ED. L'objectif le plus commun dans l'AC-OPF est la minimisation des coûts de production, avec ou sans considération des pertes du système. De cette façon, l'AC-OPF étend le problème de l'ED classique: l'ED contrôle seulement les unités génération tandis que l'OPF contrôle aussi les flux d'énergie dans le système [11]. Si nous cherchons à minimiser le coût de production, la fonction globale du système représente la somme du coût de production à chaque noeud, qui est souvent estimé à l'aide des courbes du second degré [12, 13]. Les contraintes considérées sont aussi quadratiques, le problème d'optimisation est ainsi non convexe.

Une des simplifications les plus courantes de l'OPF est le DC-OPF [14]. Le DC-OPF a été utilisé par de nombreux utilitaires pour la prévision des prix et la planification du système [15, 16]. Il gagne de plus en plus d'intérêt en raison de l'utilisation et le développement des technologies DC et HVDC. Le DC-OPF est utilisé par les opérateurs de système indépendant (ISO) dans l'Amérique du Nord pour le fonctionnement au jour le jour. Il peut être considéré comme un problème d'optimisation convexe avec des contraintes linéaires. Principalement on l'utilise en cas d'absence de données de contrôle de tension Var fiables, ou en raison d'un besoin de compatibilité croisée entre deux ou plusieurs applications connexes [14–17].

Ces deux modèles sont considérés comme des modèles de l'OPF sans contingence. Ils représentent un cas de base et ne garantissent le respect des contraintes d'opértion du réseau que dans les conditions normales de fonctionnement. L'intégration des concernes de sécurité dans l'un d'eux conduit au problème de SC-OPF.

Le SC-OPF considère les pannes qui peuvent survenir sur les lignes de transmission et/ou installations de génération [18]. Ce modèle comprend plusieurs cas d'urgence outre que le cas de base. Le problème tend à être de grande dimension, selon le type et le nombre de contingences considérées. Pour chaque contingence considérée, nous devons tenir compte d'une nouvelle série de contraintes. En outre, un ensemble de variables additionnel, les variables du cas d'urgence, doit être considéré avec quelques contraintes d'inégalité reliant ces variables aux variables de base.

Gestion des réseaux cellulaire

L'augmentation de la demande des utilisateurs mobiles [19] dessine des discussions à la fois dans l'industrie et dans le milieu acdémique, poussant l'infrastructure mobile actuelle à évoluer vers la prochaine génération (5G) des réseaux cellulaires mobiles [20]. Une des solutions pour satisfaire cette demande et de décharger le backhaul et de stocker de manière proactive le contenu des utilisateurs à la périphérie du réseau mobile, soit dans les stations de base (SBS) ou les terminaux d'utilisateurs

Notre contribution principale dans ce domaine est de formuler la politique de stockage comme un problème d'optimisation convexe et de fournir un algorithme distribué mis en oeuvre à chaque SBS. Plus précisément, nous définissons une fonction de coût convexe globale comme la somme de fonctions convexes locales qui dépendent de la demande et de la topologie du réseau. Des contraintes linéaires sont données sur le partage d'un fichier et sa disponibilité; et sur la capacité de stockage de chaque SBS. Pour résoudre ce problème, nous adoptons une approche similaire à 5 où nous avons utilisé ADMM. Ceci permet à chaque SBS de résoudre un sous-problème de complexité réduite obtenu en utilisant cet algorithme

En bref, la motivation d'utiliser une telle approche distribuée est à 1) éviter la surcharge de communication entre SBS et de le CS qui est en charge du mécanisme de décision, et 2) répartir l'opération de calcul entre les SBS.

Dans cette partie, nous considérons le problème de la mise en cache distribuée dans les réseaux cellulaires de prochaine génération 5G où les stations de base de petites densément déployées sont capables de stocker et de livrer le contenu des utilisateurs. En particulier, nous formulons la politique d'allocation de cache optimale comme un problème d'optimisation convexe où un sous-ensemble de SBS ont leur propre fonction locale de coût qui capte l'aspect de l'utilisation du backhaul en termes de bande passante et un ensemble de paramètres du réseau et les contraintes de stockage ainsi que les dépenses sur le partage du contenu avec d'autres SBS. Dans ce contexte, une SBS peut privilégier l'option de partage du contenus avec ses voisins lorsque le coût du partage est moins que le coût d'aller chercher le fichier du CS. Compte tenu du fait que la coordination est impliquée entre SBS et en raison de la capacité de l'ADMM pour résoudre un tel problème d'optimisation compliquée d'une manière simplifiée, nous fournissons une solution distribuée pour le problème de stockage en utilisant ADMM. L'application de l'ADMM dans un tel contexte converge et la solution optimale pour le problème de caching est atteinte.

Plan de la thèse et contributions

Deux problèmes différents sont ainsi élaborés dans cette thèse, naturellement le rapport est divisé en deux parties le long des chapitres suivants.

Gestion distribué de l'énergie dans les réseaux électriques

Dans le chapitre 3 nous commençons par rappeler la transition du réseau électrique classique vers le Smart Grid. On énonce l'intervention faite sur les différents niveaux composant ce réseau. Certaines améliorations, en particulier la partie relative au gestion du réseau, doivent leur existence au système de gestion d'énergie (EMS) qui traite le problème des flux de puissance optimale (*Otpimal power flow, OPF*) et ses différents modèles. Nous nous concentrons ici sur la version à courant continu du problème, le DC-OPF, une version linéaire du problème de l'OPF. Nous présentons quelques-unes des méthodes utilisées pour résoudre ce problème et on cite le travail qui a été effectué dans la littérature connexe existante.

Nous détaillons le problème DC-OPF au chapitre 4. Nous introduisons le problème distribué du partage de la production (*Ditributed Production Sharing, DPS*) et le modèle du réseau interconnecté. Nous utilisons la formulation DPS comme un modèle générique auquel le problème DC-OPF peut être projeté comme une simple instanciation. Ainsi, nous nous concentrons notre attention au réseau électrique, nous introduisons les notations de base qui nous permettent de formuler les équations de l'OPF. En utilisant ces équations ainsi que le but de minimiser le coût de fonctionnement économique du réseau électrique; nous formulons le problème DC-OPF. Après avoir remarqué que ce problème peut être représenté en utilisant la formulation DPS, nous passons notre intérêt à résoudre le problème DPS.

En raison des inconvénients de résoudre un tel problème d'une manière centralisée, nous cherchons maintenant à le résoudre distributivement. Du fait des limites de performance des méthodes existantes, nous tournons notre attention vers l'ADMM dans le chapitre 5. Nous commençons par introduire quelques notions de la théorie des opérateurs monotones qui seront nécessaires à la compréhension de la plupart des résultats développés dans le cadre de ce chapitre. Ensuite, nous utilisons ces notions sur le dual du problème DPS. Ainsi un mécanisme de fractionnement, connu comme le *Douglas-Rachford Splitting Method* est utilisé avec l'algorithme de point proximal *Proximal Point Algorithm* (PPA) afin de dériver l'ADMM répartis par zone et de montrer sa convergence.

Dans le chapitre 6 nous appliquons cette méthode sur le problème DPS.

L'ADMM a été prouvée de tourner très bien dans un tel cas d'optimisation distribuée dans les réseaux interconnectés. Mais elle souffre d'être synchrone comme les autres méthodes existantes. Les agents ou processeurs ne possédent pas les mêmes tâches ni les mêmes caractéristiques techniques. Ces différences intensifient les inconvénients du comportement synchrone de la méthode. Ainsi, un système de communication asynchrone est considéré dans ce chapitre et il est prouvé à converger lorsqu'il est appliqué à un tel système.

La convergence est limitée au cas où les zones se chevauchent. A savoir ce chevauchement signifie que les zones sont obligées à partager certains noeuds. Dans les réseaux électriques cette situation peut conduire à des conflits dans la prise de décision. Ainsi, nous étendons l'application des algorithmes ADMM sychrones et asynchrones au cas où les zones ne se chevauchent pas. Dans ce cas, chaque noeaud appartient à une seule zone. Ce résultat est obtenu par l'introduction de noeuds fictifs sur les lignes d'interconnexion reliant les noeuds qui appartiennent à différents zones. Le réseau étudié dans ce cas est considéré dans un état statique. Aucin caractère aléatoire est présenté par les composantes du réseau et le partage est effectué géographiquement.

Dans le chapitre 7 nous introduisons un réseau constitué de générateurs conventionnels et des sources d'énergie renouvelables. Certains noeuds dans le réseau sont également équipés de dispositifs de stockage. L'existence d'un tel équipement fournit au réseau une flexibilité de partage de l'énergie à l'échelle du temps. Après avoir décrit le modèle dans la main, nous formulons le problème q'on résoud en utilisant les techniques de Lyapunov et ADMM.

Avec ce travail, nous arrivons au chapitre 8 de cette thèse. Dans ce chapitre, nous nous rappelons les résultats des chapitres précédents et nous fournissons des implémentations des algorithmes ADMM sur différents réseaux. On utilise les systèmes IEEE−30 et 118 bus pour la mise en oeuvre du problème DC-OPF dans les réseaux classiques avec des sources d'énergie conventionelles. Nous comparons plusieurs implémentations de l'algorithme sous de multiples scénarios afin de démontrer leurs propriétés de convergence et d'adaptabilité. Enfin, le système de test IEEE−6 Bus est utilisé pour le cas d'un réseau intégrant des unités de production renouvelables et des batteries de stockage. Nous suivons l'effet du partage dans le temps en raison de la présence de dispositifs de stockage avant de conclure ce chapitre.

Caching distribué dans les réseaux 5G

Avec le chapitre 9 on effectue la transition à la deuxième partie de cette thèse. Le stockage ou *cahcing* dans les réseaux 5G est notre sujet d'intérêt à ce point.

Nous commençons par une brève introduction où nous résumons les travaux connexes dans ce domaine. Ensuite, nous introduisons le modèle du réseau et on formule le problème de caching comme un problème d'optimisation convexe qui peut également être projeté sur le problème DPS.

Classiquement ce problème doit être résolu par le CS de manière centralisée. Nous reformulons le problème d'une manière distriuée et nous appliquons l'ADMM afin de le résoudre au niveau des noeuds avec une coordination limitée entre les stations de base en fonction de la situation considérée. Les noeuds sont autorisés à partager leur contenu, le problème tend à devenir compliqué et nécessite une modélisation sophistiquée. Nous essayons de simplifier le développement de l'application de l'ADMM en fournissant un exemple de 3 noeuds. L'algorithme que nous développons dans le présent chapitre est élaboré pour un réseau plus général ayant un nombre quelconque de noeuds et de fichiers à stocker. Le problème évolue avec le nombre de noeuds, les connexions et les contenus qui rendent l'optimisation distribuée par ADMM plus attrayante pour un tel domaine d'application.

Publications

La liste des publications au cours de cette thèse sont énumérées ci-dessous.

Papiers de Conférences

- **A. Abboud**, R. Couillet, M. Debbah and H. Siguerdidjane.
  "Asynchronous Alternating Direction Method of Multipliers Applied to The Direct-Current Optimal Power Flow Problem." In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2014, p. 7764 - 7768*

- **A. Abboud**, E. Bastug, K. Hamidouche and M. Debbah.
  "Distributed Caching in 5G Networks: An Alternating Direction Method of Multipliers Approach." In *IEEE 16th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC), 2015, p. 171 - 175*

Papiers de Journaux

- **A. Abboud**, F. Iutzeler, R. Couillet, M. Debbah and H. Siguerdidjane.

"Distributed Production-Sharing Optimization and Application to Power Grid Networks." In *IEEE Transactions on Signal and Information Processing over Networks, Accepted, 2015.*

Chapter 2

Introduction

This thesis focuses on optimization problems in networked systems comprising interconnected agents. The studied problems are usually large-scale and computationally cumbersome, though they have some characteristics that can be exploited by appropriate modeling.

Solving such problems while considering a central fusion center i.e., in a centralized fashion, is computationally intractable and sometimes not feasible nor reliable. Thus, a centralized optimization technique is prohibitive in such large networks specifically when agents are physically distant and have some privacy concerns.

Distributed optimization methods appear as an appealing solution in such cases. Depending on the studied problem and the area of application, multiple approaches can be used to achieve the desired target. Specifically, we focus in this thesis on two different areas. The first area of interest is the smart power grid. In this part, we focus on the Direct-Current optimal power flow problem. This is a fundamental problem that underlies many power systems operations and planning. We seek to find, distributively with the least resources usage, the optimal power dispatch. This solution should minimize the global generation cost of the whole interconnected system while obeying to its characteristics. Mainly, we derive an algorithm that is suitable for static and dynamic networks. This algorithm is synchronous per default. It requires coordination between the different agents. A drawback that may limit the system performances when joined together with the heterogeneous capabilities of each agent. We prove this algorithm to be applicable asynchronously. Thus, we provide an asynchronous distributed optimization method where the agents act independently.

Such a method have the flexibility to manage other problems in different areas by tuning the parameters with respect to the desired application. We study another area of application, the $5G$ networks. We seek to find

distributively the best caching vector for each agent while minimizing the back-haul links usage. We model the caching problem that we also aim at solving distributively using the same method.

Two different problems are thus studied, naturally the thesis report is divided into two parts along the following chapters.

Distributed Energy Management in Power Systems

In Chapter 3 we start by recalling the transition from the conventional power grid to the smart grid and the intervention made on different levels. Some of these improvements, specifically in the management part, owe their existence to the energy management system (EMS) that deals with the optimal power flow problem (OPF) and its different models. We focus here on the Direct-Current optimal power flow (DC-OPF) problem, a linear version of the optimal power flow problem. We introduce some of the methods used to solve such problems and cite the work done by the existing related literature.

We entail the DC-OPF problem in Chapter 4. We introduce the Distributed Production-Sharing (DPS) problem and the model of the interconnected network. We use the DPS formulation as a general model to which the DC-OPF problem can be projected. Then, switching our attention to the power grid network, we introduce the basic notations that allow us to formulate the power flow equations. Using these equations along with the objective of minimizing the economic operation cost of the power grid; we formulate the DC-OPF problem. After noticing that this problem can be cast using the DPS formulation, we switch our interest to solving the DPS problem. Due to the drawbacks of centrally solving such a problem, we now seek to solve it distributively. Drawn by the performance limitations of the existing methods, we turn our attention to the Alternation Direction Method of Multipliers (ADMM) in Chapter 5. We start by introducing some notions from the monotone operator theory necessary to the understanding of most of the results developed in the course of this chapter. Then, we use these notions on the dual of the DPS problem. Next a splitting mechanism, known as the Douglas-Rachford splitting method is used along the proximal point algorithm in order to derive the area-based distributed ADMM and show its convergence.

In Chapter 6 we apply this method to the DPS problem. ADMM has been proved to perform very well for such distributed optimization over networks but it suffers from the synchronous curse of the other existing methods. Not all the agents have the same loads nor the same technical characteristics. These differences intensify the drawbacks of the synchronous behavior of the

method. Thus, an asynchronous communication scheme is considered in this chapter and is proven to converge when applied to such networked system. The convergence is limited to the case where the areas overlap. I.e., the areas are obliged to share some nodes. In power grid networks this situation may lead to conflicts in decision making. Thus, we extend the application of the ADMM algorithms to the case where the areas do not overlap. This is achieved by introducing dummy nodes on the tie-lines connecting the nodes belonging to different areas.

The network under study is only considered in static case, no randomness is exhibited by the network components themselves and sharing is performed geographically. In Chapter 7 we introduce a network consisting of conventional generators and renewable energy sources. Some of the nodes in the network are also equipped with storage devices. The existence of such equipment provides the network with the flexibility of sharing energy across time. After describing the model in hand we formulate the problem and we solve it using Lyapunov techniques and ADMM.

With this work, we arrive to Chapter 8 of this dissertation. In this chapter we recollect the results of the previous chapters and provide implementations of the ADMM algorithms on different networks. IEEE−30 and 118 Bus test systems were used for the implementation of the DC-OPF problem in conventional networks. We compare several implementations of the algorithm under multiple scenarios to demonstrate their convergence and scalability properties. Finally, the IEEE−6 Bus test system is used for the case of a network with integration of distributed generation units. We track the effect of sharing across time due to the presence of storage devices before concluding this chapter.

Distributed Caching in 5G Networks

With Chapter 9 we transit to the second part of this dissertation. Caching in $5G$ networks is our concern at this point. We start with a brief introduction where we summarize the related works in this field. Next we introduce the network model and formulate the caching problem as a convex optimization problem that can also be projected onto the Production-Sharing scheme. Conventionally this problem has to be solved by the central scheduler in a centralized fashion. We reformulate the problem and apply ADMM in order to solve the caching problem distributively at node level with limited coordination between the agents, depending on the considered situation. The nodes are allowed to share contents, the problem tends to become complicated and requires a sophisticated modeling. We try to simplify the development of the ADMM application by providing an example of 3 nodes. The algorithm we

develop in this Chapter is elaborated for a more general network with arbitrary number of nodes and contents. The problem scales with the number of nodes, connections and contents which make the distributed optimization through ADMM more appealing for such area of application.

Publications

Some of the contributions presented in this thesis have been published in the following papers.

In proceedings

- **A. Abboud**, R. Couillet, M. Debbah and H. Siguerdidjane.
  "Asynchronous Alternating Direction Method of Multipliers Applied to The Direct-Current Optimal Power Flow Problem." In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2014, p. 7764 - 7768*
- **A. Abboud**, E. Bastug, K. Hamidouche and M. Debbah.
  "Distributed Caching in 5G Networks: An Alternating Direction Method of Multipliers Approach." In *IEEE 16th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC), 2015, p. 171 - 175*

Journal articles

- **A. Abboud**, F. Iutzeler, R. Couillet, M. Debbah and H. Siguerdidjane.
  "Distributed Production-Sharing Optimization and Application to Power Grid Networks." In *IEEE Transactions on Signal and Information Processing over Networks, Accepted, 2015.*

Chapter 3

State of the art

Introduction

Electric power was discovered over a hundred years ago, since then it has played a major role in the enhancement of our way of life. Several applications and technologies have emerged, more and more services are offered to consumers and that help keeping our society functioning and improving the economies operating [1, 2].

The construction of an interconnected system of large, central generating stations, High-Voltage Alternating-Current (HVAC) and Direct-current (HVDC) transmission lines, and lower voltage AC and DC distribution lines across the countries resulted in the creation of an international power grid. In Europe for example, the European Network of Transmission System Operators, comprises of operators of 34 countries. Several of these systems are connected using HVDC technologies that exhibit greater efficiency and allow transmission between non-synchronized AC systems with lower losses at long distances [3].

Developing and managing both the transmission and distribution systems are key factors for the success in modern societies [1, 2] and play a major role in the ability of Europe's grid networks to become a single energy market [3]. With aiming to reach the Smart Gird goal, several improvements were implemented to the power grid, and many others are under study or yet only perspectives.

We go through few of these improvements in the next section followed by a study of the economic operation of the grid which leads us to the core subject of this thesis. We motivate our work, and provide a general overview of related works.

3.1 From conventional to Smart Grid

Smart Grid, also called smart electrical/power grid is a term used to represent the future electric power grid [1, 2, 21–24]. Smart Grid is expected to bring enhancements into the current power grid on three main levels.

1. Smart infrastructure: On the one hand, the smart grid infrastructure should allow for advanced power generation, delivery, and consumption technologies. This is achieved through distributed generation by integrating renewable power generation resources such as wind or solar cells to generate more green power; and by incorporating distributed storage devices to balance the variability and intermittency of such power sources. It also allows for integration of new paradigms such as Plug-in electric vehicles and plug-in hybrid electric vehicles that feature batteries. These can be plugged in at end-user premises at charging stations, they can lead to great environmental and economic benefits. On the other hand, integration of advanced distributed metering system and communication technologies represent a key component of a smart grid. This point represents a crucial requirement for enabling intelligent decision making.

2. Smart management: Leveraging the advanced infrastructure in order to achieve advanced management control objectives, makes the grid "smarter". Operation cost reduction, supply and demand balance, emission control, and utility maximization are considered as the main management objectives to be realized for a smarter power grid. Various management tools can be adopted to achieve these goals, such as optimization, game theory, machine learning.

3. Smart protection: Providing advanced grid services is a key feature of smart grid. Insuring the system reliability and failure protection while preserving the privacy and security protection of information (metering, monitoring and measurement) and communication are important requirements for smart grid stability and availability.

Achieving these goals require the intervention at different sectors and levels (power generation, transmission and distribution; information collecting, communicating and utilization, etc.) of the multiple actors composing the network (generation and distribution facilities, consumers, markets and operations, etc) [24]. Next, we give further details regarding the management of the power generation and transmission in the power grid. We concentrate on the economic operation of the grid, mainly the case of a single energy market comprising different operators/countries.

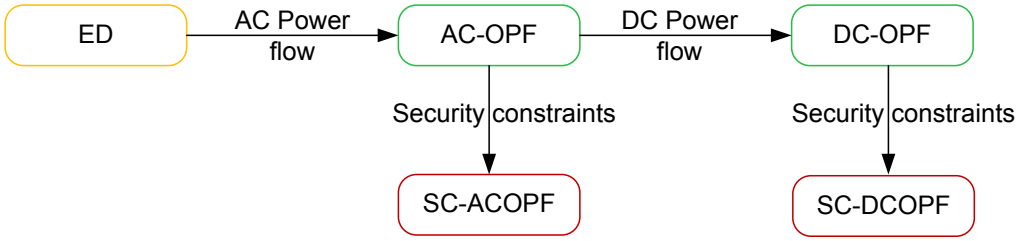

Figure 3.1: Economic operations of the grid.

3.2 Economic operation of the power grid

As previously mentioned, in the purpose of having a Smart Grid, the electrical power utility system has been subject to various improvements related to computer monitoring, communications and security. These improvements owe for their existence to the Energy Management System (EMS) and the Supervisory Control and Data Acquisition system (SCADA). EMS is used by the network operators to mainly monitor, control, and optimize the performance of the generation and/or transmission system within a certain area. SCADA system is an industrial automation control system that provides monitoring and control functions of electrical power system remote devices.

Measurements from the field are sent via SCADA system to the control center. Data flow is as follows, sensors measuring voltage magnitudes and angles and power injections at nodes or power flows and currents on lines. A state estimator converts redundant readings and other available information into an estimate of the true state of the system filtering out measurement noise [4–6]. These estimates of the network status are then used for network monitoring and operation.

EMS aims at providing reliable supply for the users at an economical cost while maintaining system's security and stability. To this end, it employs tools such as Economic Dispatch (ED) and Optimal Power Flow (OPF) [1, 4–8]. ED schedules the output of the electricity generation facilities so as to meet the system load at the lowest cost subject to transmission and operational constraints [9, 10]. It determines the most economically generated power output to serve given a certain load. ED is formulated as a convex optimization problem with an objective function summarizing the individual costs of each generator constrained by having the demand meeting the network load.

Our focus here is the OPF problem which is very important in planning efficiently the schedule of power generation. OPF models are basically categorized in i) Alternating-Current Optimal Power Flow (AC-OPF), ii)

Direct-Current Optimal Power Flow (DC-OPF), and iii) Security constrained Optimal Power Flow (SC-OPF). The transition from one model to the other is represented in Fig.3.1.

AC-OPF, or simply OPF, incorporates AC transmission network constraints into ED. The most common AC-OPF objective is the minimization of generation costs, with or without consideration of system losses. In this way, AC-OPF extends the classic economic dispatch problem: classic economic dispatch controls only which generation units to dispatch while OPF controls all power flows within the system [11]. If we aim to minimize the generation cost then, the objective function represents the sum of the generation cost at each node, which is often approximated using quadratic cost curves [12, 13]. The considered equality constraints are also quadratic thus, this is a non-convex optimization problem. The variables considered can be divided into state variables and control variables. Control variables define the system and govern the evolution of system from one state to another state. State variables describe the behavior of the system on any stage. Limits on power generation and injection have to be met, they are modeled as inequality constraints on the corresponding variables. Typical approaches rely on Karush−Kuhn−Tucker (KKT) conditions [25–27]. Several methods have been suggested for the non-convex OPF problem (explained later on), see [28–31] and references within. The drawbacks of most of the methods are that either they can only guarantee local optimum or that they are based on heuristic for which optimality cannot be proved.

One of the most common simplifications of OPF is the DC-OPF [14] formulation. The DC-OPF has been used by many utilities for price forecasting and system planning [15, 16]. It is gaining more and more interest due to the increasing use and development of DC and HVDC technologies. DC-OPF is used by several North American Independent System Operators (ISO) for day-to-day operation. It can be seen as a convex optimization problem with linear constraints. Although it is limited to the MW-oriented applications, sometimes there may be no other alternative to the use of this DC model. Mainly in the absence of reliable voltage-VAr control data, or due to a need of cross-compatibility between two or more related applications. It may also exists that markets only have access to linear theory and/ or calculations techniques so they leverage under these circumstances the DC model of the OPF. Mainly when the network tends to get larger, a large volume of computing is required, this would be prohibitive using AC-OPF model [14–17].

Both of these models are considered as contingency free OPF models. They represent a base case and only guarantees that the operating constraints are satisfied only under normal operating conditions. Incorporating security concerns in either of them leads to the SC-OPF problem. SC-OPF

considers outages that may occur in transmission lines and/ or generation plants [18]. This model includes several contingency cases along the base case. The problem tends to be very large, depending on the type and number of contingencies considered. For each contingency considered, we have to account for a new set of equality and inequality constraints. Additionally, another set of variables, the contingency case variables, has to be considered along with some inequality constraints linking these variables to the base variables.

We recall the main methods used for OPF and DC-OPF in the next section and the related work in the final section. We try to mainly focus on the DC-OPF model.

3.3 OPF and DC-OPF methods

Various approaches have been taken to solve power system OPF and DC-OPF problems, references [30–38] identify the most common conventional optimization methods advantages, drawbacks, and application. As for the artificial intelligence based methods, we can cite the fuzzy logic approach [39–41], the artificial neural network method [42] and the particle swarm optimization technique [43–47]. In this thesis, we focus on solving the power flow problem using conventional methods. The majority of the conventional techniques discussed in the literature falls under the following categories [30–38]:

1. Gradient methods: use the 1^{st} order derivative vector of the objective function to determine improving directions for the solution in iterative steps. Gradient methods are considered slow compared to higher order methods.

2. Simplex Method: Considered as the most well-known and robust formal optimization method for Linear programming (LP). It tests adjacent vertices of the feasible set in sequence so that at each new vertex the objective function improves or remains unchanged.

3. Sequential Linear Programming (SLP): Extends the LP method and solves problems with nonlinear characteristics via a series of linear approximations. SLP is known to easily handle all types of continuous constraints.

4. Sequential Quadratic Programming (SQP): Solves a series of QP problems that converge to the optimal solution of the original problem. At each iteration, a quadratic program (QP) that approximates the behavior of the problem about a particular operating point is generated.

The obtained optimal solution of the QP is then used as the starting point for the next iteration, and the process is repeated to convergence.

5. Newton's method: Represents a 2^{nd} order derivative method, usually applied to unconstrained problems with a guaranteed quadratic convergence. Both OPF and DC-OPF have equality constraints, this is why Lagrangian function has to be utilized. Inequality constraints are either treated as equality constraints or omitted. To enhance the convergence speed, other methods used an augmented version of the Lagrangian.

6. Interior Point methods (IPMs): Solve the problem or the KKT conditions of the problem by applying a variant of Newtons method to a sequence of equality constrained problems, or to a sequence of modified versions of the KKT conditions. A key feature of IPMs is that they constrain the search to the feasible region by introducing barrier terms to the augmented objective function or by manipulating the required KKT conditions.

We provide in the next section a general overview of the related works that use these methods.

3.4 Related work

The electric power system tends to be a very large scale system, it may span multiple states, countries, and even continents. The responsibility for the control of such a system is therefore shared among multiple entities. In the smart grid paradigm, more renewable energy sources will be incorporated into the system, especially in distribution networks, and the problem size will also grow tremendously.

One way to solve the DC-OPF problem is for each party involved to share their problem data with a central station that solves the problem. Most real world OPF applications are related with large scale networks, and therefore direct applications of centralized methods can be ineffective. For this reason we are inspired to go for a distributed algorithm where the network is seen as a cluster of connected sub-networks. The network can be seen as a set of nodes and a set of edges connecting them. It is interesting to note that most real world examples of power networks are sparse, each node in the network is only connected to few others through power lines [48]. In this context, we can consider the existence of a processor per sub-network (also called area in this work), or even a processor per network's node. The optimal is to have these processors or agents performing local updates using only their private

cost function. They can also communicate locally with directly connected neighbors using the edges of the underlying communication graph. The DC-OPF represented as a convex optimization problem with linear constraints has been a subject of study for many years now. In the following, we describe several techniques that has been used to solve distributively the DC-OPF problems.

Linearized Augmented Lagrangian [49, 50] approaches can be found in references [51,52]. In these works, the authors replaced the constrained optimization problem by a series of unconstrained problems. Using the Auxiliary problem principle (APP) [53,54], the problem is parallelized by decomposing it into regions. The variables corresponding to border nodes are duplicated, and coupling constraints linking these variables are imposed. Additionally, quadratic penalty terms are added to the objective function. Only inter-regional communication, related to these border nodes, is required. The overall Communication overhead will be smaller than the centralized solution [51,55]. But the problem became a little bit complicated and synchronization is required.

Approximate Newton directions [56], can be seen in the work of Conejo et al. [57]. In this work, the authors solve the problem using an approximation to the KKT system of equations and a preconditioned conjugate gradient procedure to guarantee convergence. A linear system needs to be solved at the central controller, and convergence is not always guaranteed [58]. This method can be also seen as a particular case of the Lagrangian relaxation technique.

IPMs can be found in the papers [59–61] and the references within. A well-known implementation of an interior point method to the optimal reactive dispatch problem is described in [60]. This technique is shown to be very effective to relatively large scale problems as the number of required iterations is not very sensitive to network size or number of control variables.

These methods are however computationally cumbersome [62, 63]. They share the same synchronization that ties the different areas and demand strong coordination. Each area optimizes its variables in every iteration and communicates with its adjacent areas before moving to another iteration. The synchronous burden tends to cause degradation in their performance mainly because of the inherent communication overhead and the idle processing time [64]. Among those methods, there is a technique known as the Alternating Direction Method of Multipliers (ADMM) [65–69]. While sharing the same performance limitations, ADMM has the advantage of converging faster than the aforementioned methods [68, 70]. It is easier to be implemented centrally and distributively, and requires less time and computational capacity at each processor. ADMM also relies on the augmented

Lagrangian to improve the convergence rate. The structure of the iterations in this method makes their distribution among the existing processors easier than the previously mentioned methods [70].

ADMM has gained a big interest lately, multiple applications to ADMM can be found in [58,71–76] and the references within. In [71], the authors aim at optimizing the power grid monitoring infrastructure by treating the problem of power system state estimation in a network of connected areas. The linearized problem in solved distributively using ADMM. Few minimal data exchanges were required between the neighboring areas in order to converge to the optimal solution. As for its implementation to OPF problems, reference [76] used Semidefinite programming (SDP) relaxation to convert OPF into a convex problem then ADMM was applied. In the OPF context, ADMM ensures scalability with respect to the network size, robustness to communication outages and failures, while preserving data privacy and integrity. Tomaso Erseghe, presented in [58] a fully distributed scalable application of ADMM with no need to a central coordinator, but clock synchronization is always required.

Recently [77] proposed a fully asynchronous distributed version of the ADMM that is applied sequentially by randomly selected area of the network. The problem was tuned for the find the consensus on the minimizer of the aggregate cost of the nodes. The union of these areas covers the whole network. All of them must overlap i.e., they have to share some nodes. Despite the synchronicity, by relying on a careful analysis of the proximal splitting method [68] of which the ADMM is a special instance, it is proved that such an algorithm converges. The asynchronous ADMM algorithm formulated in [77] was not designed for problems resembling to the DC-OPF. It only considers unconstrained consensus problems where all the nodes share the same scalar optimization unconstrained variable. Additionally, the division of the network has to be done using an overlapping architecture. A node sharing is not feasible in reality. It requires multiple authorities taking action on the same generation plant for example.

Chapter 4

DC-OPF as a distributed production sharing problem

4.1 Introduction

Focusing on efficiently planning the schedule of the power generation and sharing in the power grid, the EMS employs the DC-OPF problem. This problem can be formulated as convex optimization problem with linear equality, inequality and box constraints. The equality constraints represent the DC power flow equations that has to be met in order for the solution to be applicable. The inequality constraints are relative to the thermal line limits. They constrain the sharing capacities between the directly connected nodes. As for the box constraints, we have the lower and upper bounds on the generated power; and the feasible values for the voltage phase angle of the buses. In this chapter we formulate the DC-OPF problem and project it into a general model that can compass it along with other convex problems. We call this problem the Production-Sharing problem. Then, following the need to distribute the computations over different processors, we motivate a distributed formulation of this Production-Sharing problem.

4.2 Production-Sharing Problem

4.2.1 Network Model

Consider a group of N agents/nodes connected by an undirected connected graph $G = (V, E)$, where $V = \{1, ..., N\}$ is the set of nodes and E is the set of bidirectional edges/branches. The nodes can be considered as agents in many applications, we will use these two terms interchangeably. Nodes

i and j are considered as neighbors if $\{i, j\} \in E$, the link between them is denoted by $i \sim j$. Assume the neighbors of node j are identified through the set $\mathcal{N}_j = \{i \in V : j \sim i\}$.

We introduce symmetric edge weights $w_{i,j} = w_{j,i}$ such that $w_{i,j} > 0$ if and only if $\{i, j\} \in E$ and 0 elsewhere. The matrix $\mathbf{W}$ defined by the entries $\{w_{i,j}\}$ is called the weighted adjacency matrix. The diagonal matrix $\mathbf{D}$ such that $D_{i,i} = \sum_{j \in V} w_{i,j}$ is called the degree matrix, and $\mathbf{L} = \mathbf{D} - \mathbf{W}$ is called the Laplacian matrix of the graph. Let $\mathbf{M}$ be the weighted branch-bus incidence matrix. Each non-zero element in $\mathbf{M}$ corresponds to the sharing ability between the corresponding nodes and the weight of such connection. Let $e \hat{=} \{i, j\} \in E$, then we suppose $M_{ei} = w_{i,j}$, $M_{ej} = -w_{i,j}$ and $M_{ek} = 0, \forall\, k \neq \{i, j\}$.

Each agent $j \in V$ produces and/or consumes some resources (typically electric power in the power grid case) represented by a real value, and the agents can exchange these resources using the links of the aforementioned graph. More precisely, each agent has a production ability in the form of an interval. This range is reduced to 0 when the node does not contribute in resource production.

4.2.2 Formulation of the Production-Sharing Problem

Assume that, at each node j, a local cost is associated to active power production through a function depending on the quantity produced (which will be assumed convex in the following).

The natural problem ensuing is thus to minimize the total production cost under the constraints that:

i) The demands are satisfied.

ii) The network is able to dispatch correctly the resource.

This last point can take multiple forms depending on the resource type and will be made explicit when considering the DC-OPF problem next. However, we will write it generically as *"the vector of the differences between production and consumption at each agent belongs to the span of the Laplacian matrix"*.

We write this problem formally as a convex optimization problem[1] with linear constraints:

[1] the introduction of the excess variable $\boldsymbol{y}$ is not mandatory but it is kept for clarity purposes.

Production-Sharing Problem

$$\begin{aligned} &\min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}\in\mathbb{R}^N} \quad \textstyle\sum_{j\in V} f_j(x_j) \\ &\text{subject to} \quad \begin{cases} \boldsymbol{y} = \boldsymbol{x} - \boldsymbol{d} \\ \boldsymbol{y} = \mathbf{L}\boldsymbol{z} \\ \underline{\boldsymbol{x}} \leq \boldsymbol{x} \leq \overline{\boldsymbol{x}} \\ \underline{\boldsymbol{p}} \leq \mathbf{M}\boldsymbol{z} \leq \overline{\boldsymbol{p}} \end{cases} \end{aligned} \tag{4.1}$$

where, each agent j has a demand d_j, a resource production x_j in a certain range $[\underline{x_j}, \overline{x_j}]$ at a price given by the local cost function f_j. $\boldsymbol{x}$ is the resource production vector, $\boldsymbol{d}$ is the demand vector, $\mathbf{L}$ is the Laplacian of graph G, $\boldsymbol{y}$ and $\boldsymbol{z}$ are the intermediate vectors casting the sharing between the connected nodes, and finally $\mathbf{M}$ is the weighted branch-bus incidence matrix.

The last constraint on the production sharing is closely related to the network characteristics and the studied problem. Depending on the considered problem, this constraint can be neglected or taken in consideration. If neglected, the problem in hand is designated as unlimited Production-Sharing problem. In this case, the nodes can share as much production as they wish with no constraints limiting these exchanges.

Unlimited Production-Sharing Problem

$$\begin{aligned} &\min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}\in\mathbb{R}^N} \quad \textstyle\sum_{j\in V} f_j(x_j) \\ &\text{subject to} \quad \begin{cases} \boldsymbol{y} = \boldsymbol{x} - \boldsymbol{d} \\ \boldsymbol{y} = \mathbf{L}\boldsymbol{z} \\ \underline{\boldsymbol{x}} \leq \boldsymbol{x} \leq \overline{\boldsymbol{x}} \end{cases} \end{aligned} \tag{4.2}$$

A problem with a similar formulation to the Unlimited Production-Sharing Problem was treated by the authors in [70]. In this work we will focus on the Production-Sharing problem. All the results obtained for this latter problem can be directly applied to the unlimited Production-Sharing problem after relaxing the limits on $\boldsymbol{z}$.

In order to be able to perform meaningful derivations, we will make the following standard assumption.

Assumption 1. *The functions $\{f_j\}_{j\in V}$ are convex, proper, lower semi-continuous and the graph G is undirected and connected. Furthermore, the set of variables minimizing Problem (4.1) is non-empty.*

A function f_j is convex if and only if its *epigraph* defined as

$$epi\, f = \{(x_j, c) \in \mathbb{R} \times \mathbb{R} : f_j(x_j) \leq c\},$$

is a convex set.

Additionally, f_j is closed if its *epigraph* is also closed. Note that f_j is proper if its effective domain $dom\ f_j \neq \emptyset$. The effective domain of f_j represents the set of points for which it takes finite values

$$dom\, f_j = \{x_j \in \mathbb{R} : f_j(x_j) < +\infty\}.$$

Thus, f_j is proper if

$$\exists\, x_j \in \mathbb{R} \text{ such that } f_j(x_j) < +\infty; \text{ and } \forall y_j \in \mathbb{R},\ f_j(y_j) > -\infty$$

A simple example: *Consider a complete graph with unit edge weights and say that each of the N agents has a unit demand ($\boldsymbol{d} = [1, ..., 1]^T$). Because the graph is complete $\mathbf{L} = N\mathbf{I} - \mathbf{J}$ where $\mathbf{I}$ is the identity matrix and $\mathbf{J}$ is filled with ones. This means that the second condition of the above problem simply translates to* "the mean of the resource production vector $\boldsymbol{x}$ is one". *Now, assume that half of the agents cannot produce any resource (for them, $\underline{x_j} = \overline{x_j} = 0$ thus $x_j = 0$) while the other half can produce any quantity x ($\underline{x_j} = 0, \overline{x_j} = +\infty$) with cost ax (for every producer j, $f_j(x_j) = ax_j$ and $a > 0$); it is immediate to see that each producer will produce the same quantity (as they are indistinguishable in the problem with a complete unweighted graph) and this quantity must be 2 (as only half of the agents produce and the mean of the production vector has to be one).*

Additional variables, equality and inequality constraints can be easily incorporated to the previous Production-Sharing problem. Inequality constraints linking different variables are replaced by equality constraints via the use of slack variables. Additional cost on a certain variable can also be incorporated to the existing cost function (it should comply with the previous assumption). All of these modifications do not interfere with the results of the next chapters, they induce no complexity as the algorithm can adapt to such requirements.

Now, let us see how this problem can encompass problems encountered in power networks such as the DC-OPF.

4.3 DC-OPF as a Production-Sharing Problem on a Graph

4.3.1 Power grid modeling

A power network consists of a set of agents/buses V and a set of edges E linking these nodes. Each agent $j \in V$ can *generate* a power p_j^G with a *cost* $f_j(p_j^G)$ in its generation ability range $[\underline{p_j^G}, \overline{p_j^G}]$ (possibly reduced to $\{0\}$ if the agent is not a generator) and has a power demand p_j^D [2, 78, 79]. Let $|v_j^o|^2$ and θ_j be the voltage magnitude and phase at node j, and i_j^c be the current injected at the same nod.

Using Kirchoff's Current Law (KCL)[3] and Ohm's Law [4], we obtain a linear relation between the current injected i_j^c at node j and the voltage angle of the nodes in its neighborhood set $\mathcal{N}_j$,

$$i_j^c = \sum_{i \in \mathcal{N}_j} y_{ji}(v_j^o - v_i^o)$$

where, $y_{ij} = g_{ij} + b_{ij}$ is the circuit admittance between nodes j and i. Note that the circuit admittance is the same when seen from node i or j perspective $y_{ij} = y_{ji}$. All the circuits are supposed to have purely inductive series components thus, susceptance b_{ij} is negative.

Writing this equation for all the nodes of the network, we obtain:

$$\boldsymbol{i}^c = \mathbf{Y}\boldsymbol{v}^{\boldsymbol{o}}$$

where, $\boldsymbol{i}^c$ is the vector of injected current, $\mathbf{Y}$ is the admittance matrix and $\boldsymbol{v}^{\boldsymbol{o}}$ is the nodal voltage vector. The admittance matrix $\mathbf{Y}$ is symmetric, i.e., $Y_{ij} = Y_{ji} = -y_{ij} = -y_{ji}$, and $Y_{jj} = \sum_{i \sim j} y_{ij}$[5].

4.3.2 Alternating Current to Direct Current power flow equations

Line $i \sim j$ is characterized by its resistance r_{ij} (in *ohms*) and its reactance x_{ij} (in *ohms*). As previously mentioned, the admittance of the circuit $i \sim j$

[2] All the quantities are assumed to be in per unit.

[3] KCL requires that each of the current injections be equal to the sum of the currents flowing out of the bus and into the lines connecting the bus to other buses.

[4] Ohm's Law states that the current through a conductor between two points is directly proportional to the voltage difference across the two points.

[5] The admittance-to-ground y_j at node j is neglected.

is given by

$$y_{ij} = g_{ij} + jb_{ij},$$

where,

$$g_{ij} = \frac{r_{ij}}{x_{ij}^2 + r_{ij}^2}$$

and

$$b_{ij} = \frac{-x_{ij}}{x_{ij}^2 + r_{ij}^2}.$$

Each couple of nodes can exchange their resources using the power line linking them. Let us consider the AC real and reactive power flow equations at node j:

$$p_j = \sum_{i \in \mathcal{N}_j} |v_j||v_i|(G_{ji}\cos(\theta_j - \theta_i) + B_{ji}\sin(\theta_j - \theta_i) \tag{4.3}$$

$$q_j = \sum_{i \in \mathcal{N}_j} |v_j||v_i|(G_{ji}\sin(\theta_j - \theta_i) - B_{ji}\cos(\theta_j - \theta_i) \tag{4.4}$$

where, p_j and q_j are the real and reactive power flow at node j respectively, G_{ij} and B_{ij} are the real and imaginary parts of Y_{ij} respectively.

Many commercial and industry OPF formulations use the DC power flow equations instead of AC power flow [80, 81]. This formulation allows the development of a fully linear set of constraints.

The DC version of the power flow equations can be derived using the following three basic considerations.

- The lines are supposed to be purely inductive i.e., $r_{ij} \ll x_{ij}$. Thus, we can approximate $g_{ij} = 0$ and $b_{ij} = \frac{-1}{x_{ij}}$. Susceptance b_{ij} is negative and all the real parts of the admittance matrix $\boldsymbol{Y}$ will be equal to zero.
- The angle separation across any circuit $i \sim j$ is small, that is $|\theta_j - \theta_i| < \frac{\pi}{12}$. Thus, $\cos(\theta_j - \theta_i) \approx 1$ and $\sin(\theta_j - \theta_i) \approx (\theta_j - \theta_i)$.
- In the per-unit system, the numerical values of voltage magnitudes $|v_j|$ and $|v_i|$ are assumed to be unity.

Using these considerations, the reactive power flow can be ignored and the real power flow equation, known as the DC power flow equation, reduces to:

$$p_j = \sum_{i \in \mathcal{N}_j} B_{ji}(\theta_j - \theta_i) \tag{4.5}$$

This DC power flow equation only calculates real (MW) flows on network's lines and gives no answers to what happens to voltage magnitudes or reactive (MVAR) flows.

4.3.3 DC power network model

The network of N agents are linked by transmission lines E, and $G = (V, E)$ forms a connected undirected graph. We start by introducing the constraints that has to be taken into consideration at each node in the network. A vectorial presentation complying with the Production-Sharing problem is then given and followed by the formulation of the DC-OPF problem.

- Node local formulation:
  - The resource production at a certain node is limited, the real power production box constraint for each node $j \in V$:

$$\underline{p_j^G} \leq p_j^G \leq \overline{p_j^G}. \tag{4.6}$$

  - In the DC model, the flow of real power along each line $i \sim j$ is computed using $B_{ji}(\theta_j - \theta_i)$. The real power thermal box constraint for each branch $i \sim j$ is given by:

$$\underline{p_{ij}} \leq B_{ji}(\theta_j - \theta_i) \leq \overline{p_{ij}}. \tag{4.7}$$

    Normally, $\underline{p_{ij}} = -\overline{p_{ij}}$. Let $p_{ij} \hat{=} B_{ji}(\theta_j - \theta_i)$. Thus, this constraint on transmitted power can be rewritten as:

$$|p_{ij}| \leq \overline{p_{ij}}. \tag{4.8}$$

  - We introduce the notion of *excess power* p_j^E of node j to designate the power injected (or withdrawn) at node $j \in V$ (4.5). The real power balance constraint for each node $j \in V$ can be written as:

$$p_j^E = p_j^G - p_j^D. \tag{4.9}$$

  - As previously explained, KCL and the DC model conditions impose the excess power (which is possibly negative) to be transmitted through the adjacent transmission lines. Formally, for every agent j, this sums up to:

$$p_j^E = \sum_{i \sim j} B_{ji}(\theta_j - \theta_i). \tag{4.10}$$

- Global vectorial presentation:

– Let $\mathbf{M}$ be the weighted branch-bus incidence matrix. Then, the power flow box constraints for all the branches can be regrouped as:

$$|\mathbf{M}\boldsymbol{\theta}| \leq \overline{\boldsymbol{p}}, \tag{4.11}$$

where, $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_N)$ is the phase vector and $\overline{\boldsymbol{p}}$ is the vector of edges' thermal limits.

– Let $\boldsymbol{p}^E = (p_1^E, \ldots, p_N^E)$ be the the power injection vector, $\boldsymbol{p}^G = (p_1^G, \ldots, p_N^G)$ be the power generation vector and $\boldsymbol{p}^D = (p_1^D, \ldots, p_N^D)$ be the power demand vector. The vectorial presentation of the real power balance constraint:

$$\boldsymbol{p}^E = \boldsymbol{p}^G - \boldsymbol{p}^D. \tag{4.12}$$

– The real power flow condition can also be rewritten in a vectorial manner as the following multivariate power model:

$$\boldsymbol{p}^E = \mathbf{L}^B\boldsymbol{\theta}, \tag{4.13}$$

where $\mathbf{L}^B$ is the Laplacian matrix obtained from the weighted adjacency matrix $\mathbf{B}$. $\mathbf{B}$ is symmetric and positive semi-definite with $B_{jj} = 0$ and $B_{ji} = -b_{ji}$[6].

Using these vectorial presentation of the constraints, a first formulation of the DC-OPF Production-Sharing problem is given as follows.

DC-OPF Production-Sharing Problem

$$\begin{array}{ll} \min\limits_{\boldsymbol{p}^G, \boldsymbol{p}^E, \boldsymbol{\theta} \in \mathbb{R}^N} & \sum_{j \in V} f_j(p_j^G) \\ \text{subject to} & \begin{cases} \boldsymbol{p}^E = \boldsymbol{p}^G - \boldsymbol{p}^D \\ \boldsymbol{p}^E = \mathbf{L}^B\boldsymbol{\theta} \\ \underline{\boldsymbol{p}^G} \leq \boldsymbol{p}^G \leq \overline{\boldsymbol{p}^G} \\ |\mathbf{M}\boldsymbol{\theta}| \leq \overline{\boldsymbol{p}} \end{cases} \end{array} \tag{4.14}$$

It is straightforward to see that the above problem has the same form as Problem (4.1). It is important to notice that the matrix $\mathbf{B}$ (through its Laplacian) suffices to structure the generated power vector to comply with electrical equality constraints across the power network.

[6] B_{ji} corresponds to the imaginary part of the entry Y_{ji} of admittance matrix $\mathbf{Y}$.

4.4 Centralized Production-Sharing

The Production-Sharing problem introduced previously is essentially centralized. Indeed, while the underlying graph is presented through the Laplacian matrix, the condition $\boldsymbol{y} = \mathbf{L}\boldsymbol{z}$ on the transmission of the excess production implies a coordinated action of the whole network.

The traditional way to deal with such Production-Sharing problem, in a large network of connected agents, relies on the existence of a central fusion center. This fusion center makes a decision with the aggregated data collected from all the agents, then transmits each value to its corresponding node in the network.

A large-scale centralized Production-Sharing problem suffers from several technical and institutional drawbacks. On the one hand, this centralized optimization method is prohibitive in systems that spread over several physically distant devices due to the communication burden and the delays induced by the required computations and communication [21]. On the other hand, the privacy concerns of the end users and/or authorities (in case of an environment with multiple authorities), and the security issues related to having only one central coordinator (e.g., single point of failure, bottleneck and agreement in taking decisions) make it impossible to efficiently implement the centralized optimization [82].

In order to decrease the communication cost and the computation capacity requirements at the fusion center, it is advisable to distribute the computation load over different devices. A goal that can be easily met, by leveraging on the presence of independent processors in each component of the power grid. Thus, a distributed implementation of the Production-Sharing problem is preferable and practically applicable.

4.5 Distributed Production-Sharing (DPS)

Operating in a distributed multiprocessor environment can potentially greatly increase the available computational capacity and decrease the communication burden. Controlling a power system requires high-speed calculation which can be obtained using existing multiprocessors allocated to subsystems.

In this section, we design an equivalent problem where the network condition is split into overlapping sub-graphs and add an additional indicator function in order to ensure the equivalence with the former problem. Thus, we convert the centralized Production-Sharing problem into a distributed one.

4.5.1 Distributed Formulation of the Problem

Firstly , the relation between $\boldsymbol{y}$ and $\boldsymbol{z}$, $\boldsymbol{y} = \mathbf{L}\boldsymbol{z}$ can be substituted in $\boldsymbol{y} = \boldsymbol{x} - \boldsymbol{d}$. Problem (4.1) is reformulated as follows:

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}\in\mathbb{R}^N} \quad & \textstyle\sum_{j\in V} f_j(x_j) \\ \text{subject to} \quad & \begin{cases} \mathbf{L}\boldsymbol{z} = \boldsymbol{x} - \boldsymbol{d} \\ \underline{\boldsymbol{x}} \leq \boldsymbol{x} \leq \overline{\boldsymbol{x}} \\ \underline{\boldsymbol{p}} \leq \mathbf{M}\boldsymbol{z} \leq \overline{\boldsymbol{p}} \end{cases} \end{aligned} \tag{4.15}$$

Secondly, if we consider the boundaries on the sharing operation i.e., $\underline{\boldsymbol{p}} \leq \mathbf{M}\boldsymbol{z} \leq \overline{\boldsymbol{p}}$. Then, we can modify these constraints by mean of slack variables in order to obtain a system with only equality constraints. Let $\boldsymbol{s_1} = (s_{1,1}, \ldots, _{1,m})$ and $\boldsymbol{s_2} = (s_{2,1}, \ldots, s_{2,m})$ be the two set used for this purpose. The number of these inequality constraints is $m = |E|$, the slack variables are chosen such that $s_{1,i} \geq 0$ and $s_{1,i} \geq 0$, $\forall i$. This leads to the following problem:[7]

$$\begin{aligned} \min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{z}\in\mathbb{R}^N} \quad & \textstyle\sum_{j\in V} f_j(x_j) \\ \text{subject to} \quad & \begin{cases} \boldsymbol{y} = \boldsymbol{x} - \boldsymbol{d} \\ \boldsymbol{y} = \mathbf{L}\boldsymbol{z} \\ \underline{\boldsymbol{x}} \leq \boldsymbol{x} \leq \overline{\boldsymbol{x}} \\ \mathbf{M}\boldsymbol{z} + \boldsymbol{s_1} = \overline{\boldsymbol{p}} \\ -\mathbf{M}\boldsymbol{z} + \boldsymbol{s_2} = -\underline{\boldsymbol{p}} \end{cases} \end{aligned} \tag{4.16}$$

In the DC-OPF case, $\boldsymbol{x}$ is used to represent the power generation vector $\boldsymbol{p}^G$, $\boldsymbol{y}$ for the power injections $\boldsymbol{p}^E$, and $\boldsymbol{z}$ to symbolize the voltage phase vector $\boldsymbol{\theta}$. Additionally, $\boldsymbol{d}$ represents the power demand vector $\boldsymbol{p}^D$.

Starting from Problem (4.16), as every agent can control its resource production (and thus its excess) as well as its network intermediate vector (its phase in the DC-OPF case), it is convenient to stack these variables into a vector of size $2N + 2m$, $\boldsymbol{u} = [\boldsymbol{x}; \boldsymbol{z}; \boldsymbol{s_1}; \boldsymbol{s_2}]$.

With this new vector $\boldsymbol{u}$ that sketches the global network state, it is useful to introduce a function F that corresponds to the function we seek to mini-

[7] We chose to relax the constraints on the maximal power sharing between connected nodes or to incorporate these constraints into the global cost function. In this case we can discard the weighted incidence matrix $\mathbf{M}$ and the corresponding clack variables. The formulation of the problem become much simpler.

mize in (4.1). F is applied to $\boldsymbol{u}$ and it encompasses the range conditions[8]:

$$\begin{array}{rccl} F: & \mathbb{R}^{2N+2m} & \longrightarrow & (-\infty, +\infty] \\ & \boldsymbol{u} = \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{z} \\ \boldsymbol{s_1} \\ \boldsymbol{s_2} \end{bmatrix} & \longmapsto & \begin{cases} \sum\limits_{j \in V} f_j(x_j) & \text{if } \forall j, x_j \in [\underline{x_j}, \overline{x_j}] \\ +\infty & \text{elsewhere} \end{cases} \end{array} \tag{4.17}$$

One can remark that, provided the functions $\{f_j\}_{j \in V}$ are convex, F is also convex. Now, the rest of the conditions can be written linearly as:

$$\underbrace{\begin{bmatrix} \mathbf{I} & \mathbf{L} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & -\mathbf{M} & \mathbf{0} & \mathbf{I} \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{z} \\ \boldsymbol{s_1} \\ \boldsymbol{s_2} \end{bmatrix}}_{\boldsymbol{u}} = \underbrace{\begin{bmatrix} \boldsymbol{d} \\ \overline{\boldsymbol{p}} \\ -\underline{\boldsymbol{p}} \end{bmatrix}}_{\tilde{\boldsymbol{d}}} \tag{4.18}$$

Hence, Problem (4.16) rewrites as

$$\begin{array}{ll} \min\limits_{\boldsymbol{u} \in \mathbb{R}^{2N+2m}} & F(\boldsymbol{u}) \\ \text{subject to} & \mathbf{A}\boldsymbol{u} = \tilde{\boldsymbol{d}} \end{array} \tag{4.19}$$

Let us recall that in our scheme, each agent j has the knowledge of its own cost function f_j, its production-related variables x_j, z_j, the j-th row/column of the Laplacian matrix $\mathbf{L}$ and the j-th column of the branch-bus incidence matrix $\mathbf{M}$.

In order to derive a distributed algorithm, a common solution is to introduce one variable per condition line per agent and an indicator function ensuring that for each condition line, the sum of the agents related variables is equal to the corresponding value in $\tilde{\boldsymbol{d}}$. Let $[\mathbf{A}]_i$ represent the i^{th} line of matrix $\mathbf{A}$. Formally, we define the matrix:

$$\tilde{\mathbf{A}} = \begin{bmatrix} \text{diag}([\mathbf{A}]_1) \\ \vdots \\ \text{diag}([\mathbf{A}]_{N+2m}) \end{bmatrix} \tag{4.20}$$

[8] In the same way we can encompass other range conditions e.g., limits on $\boldsymbol{z}$, $\boldsymbol{y}$, $\boldsymbol{s_1}$ and $\boldsymbol{s_2}$

and the indicator function[9],

$$C: \begin{array}{ccc} \mathbb{R}^{(N+2m)(2N+2m)} & \longrightarrow & (-\infty, +\infty] \\ \boldsymbol{v} & \longmapsto & \begin{cases} 0 \quad \text{if } \forall i = 1 \dots N+2m, \sum_{j=2(i-1)(N+m)+1}^{2i(N+m)} v_j = \tilde{d}_i \\ +\infty \quad \text{elsewhere} \end{cases} \end{array} \tag{4.21}$$

One can note that vector $\boldsymbol{v}$ has a lot more components than actually needed. Due to the sparsity of the matrix $\mathbf{A}$, several components in $\boldsymbol{v}$ are equal to zero and can be omitted. But, for notation simplicity, the full vector will be kept until we provide the actual algorithm derivation, where we will eliminate unnecessary components and precise the actual number of variables to be updated per agent.

Finally, we obtain the following distributed problem.

Distributed Production-Sharing Problem

$$\min_{\boldsymbol{u} \in \mathbb{R}^{2N+2m}} \quad F(\boldsymbol{u}) + C(\tilde{\mathbf{A}}\boldsymbol{u}) \tag{4.22}$$

By construction, the solutions of this problem allow to derive the solutions of the original problem as mentioned in the following lemma.

Lemma 1. *Let Assumption 1 hold. Then, the solutions of Problem (4.22) are of the form* $\boldsymbol{u}^\star = [\boldsymbol{x}^\star; \boldsymbol{z}^\star; \boldsymbol{s}_1^\star; \boldsymbol{s}_2^\star]$ *where* $(\boldsymbol{x}^\star, \boldsymbol{z}^\star)$ *is a solution of Problem (4.15).*

4.6 Conclusion

In this chapter, we provided in details how the DC-OPF problem can be obtained from the AC power flow problem. We formulated the DC-OPF as convex problem with linear equality, inequality and box constraints. We motivated a general model, the Production-Sharing problem. This model can compass the DC-OPF problem along with other convex problems. In order to cope with the need of distributing the computation process along different actors, we provided distributed formulation of this Production-Sharing problem. Next, we demonstrate how efficiently can the problem be solved using the Alternating direction method of multipliers (ADMM) and how this method can be obtained.

[9] throughout this paper, we call indicator function, a function which returns 0 when the argument is in some set C and $+\infty$ elsewhere. It is immediate to check that is C is closed and convex, then the indicator of set C is convex and lower semi-continuous.

Chapter 5

Area-based ADMM

5.1 Introduction

We consider the DPS formulation that we derived in the previous chapter. Following the work of Eckstein [69] and Iutzeler [77], we provide step by step the area-based distributed ADMM algorithm using monotone operator theory. We start with some preliminaries regarding the basics of monotone operator theory. We shift the given problem to the dual domain . After which, we cite the Proximal Point Algorithm (PPA) and we show how it can be used along the Douglas-Rachford (DR) splitting method in order to derive the Alternating Direction Method of Multipliers (ADMM) algorithm.

5.2 Background Material from Monotone Operator Theory

Take an Euclidean space set $\mathcal{Y}$. We define its power set, denoted $\mathcal{P}(\mathcal{Y}) = 2^{\mathcal{Y}}$, as the family of all subsets of $\mathcal{Y}$ including the empty set $\emptyset$ and $\mathcal{Y}$ itself. An operator $D : \mathcal{X} \to \mathcal{Y}$ maps every point $\boldsymbol{x} \in \mathcal{X}$ to a point $D\boldsymbol{x} \in \mathcal{Y}$, while a set valued operator $D : \mathcal{X} \to 2^{\mathcal{Y}}$ maps every point $\boldsymbol{x} \in \mathcal{X}$ to a subset $D\boldsymbol{x} \subset \mathcal{Y}$.

An operator D (single-valued or multi-valued), is characterized by its:

- *graph:* $\operatorname{gra}(D) = \{(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{X} \times \mathcal{Y} \mid \boldsymbol{y} \in D\boldsymbol{x}\}$;
- *domain:* $\operatorname{dom}(D) = \{\boldsymbol{x} \in \mathcal{X} | \exists\, \boldsymbol{y} \in \mathcal{Y} : \boldsymbol{y} \in D\boldsymbol{x}\}$;
- *inverse:* $D^{-1} = \{(\boldsymbol{y}, \boldsymbol{x}) \in \mathcal{Y} \times \mathcal{X} \mid (\boldsymbol{x}, \boldsymbol{y}) \in gra(D)\}$;
- *zero's set:* $\operatorname{Zer}(D) = D^{-1}0 = \{\boldsymbol{x} \in \mathcal{X} \mid 0 \in D\boldsymbol{x}\}$;

- *set of fixed points:* $\mathrm{Fix}\,(D) = \{\boldsymbol{x} \in \mathcal{X} \mid \boldsymbol{x} \in D\boldsymbol{x}\}$.

An operator $D : \mathcal{Y} \to 2^{\mathcal{Y}}$ is

- *monotone* if,
  $\langle \boldsymbol{x} - \boldsymbol{x}', \boldsymbol{y} - \boldsymbol{y}' \rangle \geq 0 \quad \forall\, (\boldsymbol{x}, \boldsymbol{y}), (\boldsymbol{x}', \boldsymbol{y}') \in gra\,(D)$;
- *maximal* if,
  $\nexists D' : \mathcal{Y} \to 2^{\mathcal{Y}}$, such that $gra\,(D) \subset gra\,(D')$;
- *non-expansive* if,
  $\|\boldsymbol{x} - \boldsymbol{x}'\| \geq \|\boldsymbol{y} - \boldsymbol{y}'\| \quad \forall\, (\boldsymbol{x}, \boldsymbol{y}), (\boldsymbol{x}', \boldsymbol{y}') \in gra\,(D)$;
- *firmly non-expansive* if,
  $\langle \boldsymbol{x} - \boldsymbol{x}', \boldsymbol{y} - \boldsymbol{y}' \rangle \geq \|\boldsymbol{y} - \boldsymbol{y}'\|^2 \quad \forall\, (\boldsymbol{x}, \boldsymbol{y}), (\boldsymbol{x}', \boldsymbol{y}') \in gra\,(D)$.

Let $\mathcal{I}$ be the identity operator, for an operator $D : \mathcal{Y} \to 2^{\mathcal{Y}}$ the following expressions are equivalent

- D is firmly non-expansive,
- $\mathcal{I} - D$ is firmly non-expansive,
- $D^{-1} - \mathcal{I}$ is maximally monotone,

Let D be an operator, the resolvent of operator D is defined as

$$\mathcal{J}_{\rho D} = (I + \rho D)^{-1}$$

From [83], [68] and [84] we can deduce the following:

Lemma 2. *An operator D is monotone if and only if $\mathcal{J}_{\rho D}$ is firmly non-expansive. Additionally, if D was maximal the, $\mathcal{J}_{\rho D}$ has a full domain.*

The sub-gradient mapping ∂h of a proper convex function $h : \mathcal{Y} \to \mathbb{R} \cup \{+\infty\}$ is a monotone operator [85]. It is given by

$$\begin{aligned} &\partial h : \mathcal{Y} \to \mathcal{Y} \\ &\partial h(x) = \{y \mid \forall x' \in \mathcal{Y}, \langle x' - x, y \rangle + h(x) \leq h(x')\}. \end{aligned} \tag{5.1}$$

Additionally, if h is closed then, the sub-gradient mapping ∂h of h is maximally monotone on $\mathcal{Y}$ [85]. Furthermore, finding the minimum of h is equivalent to finding the zero of ∂h.

5.3 Dual problem

We consider the following optimization problem:[1]

$$\begin{array}{ll} \min\limits_{\boldsymbol{u}\in\mathbb{R}^{2(N+m)}} & F(\boldsymbol{u}) \\ \text{subject to} & \mathbf{A}\boldsymbol{u} = \tilde{\boldsymbol{d}}, \end{array} \tag{5.2}$$

where, F is a convex cost function, $\boldsymbol{u} \in \mathbb{R}^{2(N+m)}$ and $\tilde{\boldsymbol{d}} \in \mathbb{R}^{N+2m}$.

From $\mathbf{A}$ we define the following matrix:

$$\tilde{\mathbf{A}} = \begin{bmatrix} \operatorname{diag}([\mathbf{A}]_1) \\ \vdots \\ \operatorname{diag}([\mathbf{A}]_{N+2m}) \end{bmatrix} \tag{5.3}$$

and the indicator function:[2][3]

$$\begin{array}{rccl} C: & \mathbb{R}^{(N+2m)(2N+2m)} & \longrightarrow & (-\infty, +\infty] \\ & \boldsymbol{v} & \longmapsto & \begin{cases} 0 & \text{if } \forall i = 1 \dots N+2m, \sum\limits_{j=2(i-1)(N+m)+1}^{2i(N+m)} v_j = \tilde{d}_i, \\ +\infty & \text{elsewhere.} \end{cases} \end{array} \tag{5.4}$$

Using $\tilde{\mathbf{A}}$ and the indicator function C, we reformulate (5.2) as follows:

$$\min_{\boldsymbol{u}\in\mathbb{R}^{3N}} \quad F(\boldsymbol{u}) + C(\tilde{\mathbf{A}}\boldsymbol{u}). \tag{5.5}$$

Let $\boldsymbol{\lambda}$ be the vector of Lagrangian multipliers associated to the new set of constraints $\tilde{\mathbf{A}}\boldsymbol{u} = \boldsymbol{v}$. $\boldsymbol{\lambda}$ and $\boldsymbol{v}$ have the same size which is equal to number of elements in $\mathbf{A}$. The dual of the reformulated problem is given by [86]:

$$\underset{\boldsymbol{\lambda}}{\text{minimize}} \quad \left\{ F^*(-\tilde{\mathbf{A}}^*\boldsymbol{\lambda}) + C^*(\boldsymbol{\lambda}) \right\}, \tag{5.6}$$

where, functions F^* and C^* are the Fenchel's conjugate of f and g respectively, i.e., $F^*(\boldsymbol{u}) = \sup\limits_{\boldsymbol{x}} \{\langle \boldsymbol{x}, \boldsymbol{u}\rangle - f(\boldsymbol{x})\}$. $\tilde{\mathbf{A}}^T$ is the transpose of $\tilde{\mathbf{A}}$.

[1] We considered these sizes for the variables and matrices in order to comply with the distributed DC-OPF Production-Sharing problem formulated in Chapter 2.

[2] Throughout this paper, we call indicator function, a function which returns 0 when the argument is in some set C and $+\infty$ elsewhere. It is immediate to check that is C is closed and convex, then the indicator of set C is convex and lower semi-continuous.

[3] When $\mathbf{A}$ is a $(N+2m) \times (2N+2m)$ matrix, then $\boldsymbol{v}$ and $\boldsymbol{\lambda}$ which have the same size are a $(N+m)(2N+2m) \times 1$ vectors. For simplicity we will only use $|\boldsymbol{v}|$ and $|\boldsymbol{\lambda}|$ to refer to their size.

By strong duality, we now seek to calculate the minimum of this dual[4]. The value minimizing it coincides with the opposite of the solution of the original problem [87, Prop 3.31]. Thus, we shift to the dual domain where finding this minimum can be achieved by utilizing the proximal point algorithm (PPA).

5.4 Proximal Point Algorithm (PPA)

We denote by

$$D = -\tilde{\mathbf{A}}\, \partial F^* \circ (-\tilde{\mathbf{A}}^T) + \partial C^* \tag{5.7}$$

the sub-gradient mapping the dual problem (5.6).

We suppose that $\boldsymbol{\lambda}$ is the minimum of this dual (5.6). Then by Fermat's rule [86, Th 16.2], $\boldsymbol{\lambda}$ is also the zero of D. Additionally, the operator D is a single valued maximal monotone operator [85]. Thus, by [87, Th 3.6] and for any $\rho > 0$, the resolvent of D defined as

$$\mathcal{J}_{\rho D} = (I + \rho D)^{-1} \tag{5.8}$$

is a single valued firmly non-expansive operator with full domain. The following Lemma is then applicable.

Lemma 3. *[83, PPA] : Given a maximal monotone operator D, such that $Zer(D) \neq \emptyset$. Then $Fix(\mathcal{J}_{\rho D})$ is a singleton and $Zer(D) = Fix(\mathcal{J}_{\rho D})$. Moreover, starting from any initial point $\boldsymbol{\zeta}^0 \in dom(D)$, $\boldsymbol{\zeta}^k \to Fix(\mathcal{J}_{\rho D})$, where $\boldsymbol{\zeta}^{k+1} = \mathcal{J}_{\rho D}(\boldsymbol{\zeta}^k)$, $k \geq 1$.*

As previously mentioned, the minimum of the dual problem (5.6) coincides with the zero of its sub-gradient mapping D. By Lemma 3, instead of searching for $\mathrm{Zer}\,(D)$, we can iteratively find $\mathrm{Fix}\,(\mathcal{J}_{\rho D})$. That is, starting from any initial point $\boldsymbol{\zeta}^0$, we iterate $\boldsymbol{\zeta}^{k+1} = \mathcal{J}_{\rho D}(\boldsymbol{\zeta}^k)$ until convergence.

Thus, PPA can be applied to the sub-gradient of the dual problem in order to find its minimum. However, we should calculate the resolvent $\mathcal{J}_{\rho D}$ which appears to be a complicated task. This results from the fact that the sub-gradient D of the dual problem comprises at the same time the global objective function and the constraints which are of heterogeneous natures. Most of the optimization methods solve this problem by splitting it [88]. For this reason, we make use of the Douglas-Rachford (DR) splitting

[4] The original dual form is $\underset{\boldsymbol{\lambda}}{\text{maximize}} - \left\{F^*(-\tilde{\mathbf{A}}^T\boldsymbol{\lambda}) + C^*(\boldsymbol{\lambda})\right\}$, for simplicity we use its opposite given by equation (5.6)

method [68,89,90] in order to separate D into components that are easier to be manipulated. This method is considered because D can be seen as

$$D = \underbrace{-\tilde{\mathbf{A}}\partial F^* \circ (-\tilde{\mathbf{A}}^T)}_{T} + \underbrace{\partial C^*}_{U} \tag{5.9}$$

where T and U are two maximal monotone operators [85].

5.5 Douglas-Rachford Splitting Method

The DR splitting method is used to find the zero of the sum of two maximal monotone operators. The structure of D which can be written as the sum of two maximal monotone operators T and U naturally calls for the DR splitting method in which the operators T and U are employed in separate steps [85].

The DR splitting operator derived by Lions and Mercier [90], on which we apply PPA, is given by

$$\mathcal{R} = \{(\boldsymbol{\nu} + \rho\boldsymbol{v}, \boldsymbol{\lambda} - \boldsymbol{\nu})\,; (\boldsymbol{\nu}, \boldsymbol{\alpha}) \in T, (\boldsymbol{\lambda}, \boldsymbol{v}) \in U \text{ and } \boldsymbol{\nu} + \rho\boldsymbol{\alpha} = \boldsymbol{\lambda} - \rho\boldsymbol{v}\}. \tag{5.10}$$

Let us rephrase the following result [90] regarding the monotonicity of $\mathcal{R}$.

Theorem 1. *Let $\rho > 0$. If T and U are monotone, then $\mathcal{R}$ is monotone; furthermore, the resolvent $\mathcal{S} \hat{=} \mathcal{J}_{\rho(T+U)} = (\mathcal{R}+\mathcal{I})^{-1}$ of $\mathcal{R}$ is firmly non-expansive. If T and U are maximal monotone operators, then $\mathcal{R}$ is also a maximal monotone operator; furthermore, $\mathcal{S}$ is firmly non-expansive and has full domain.*

Thus, since T and U are maximal monotone operators, $\mathcal{R}$ is also a maximal monotone operator [68,90]. The resolvent $\mathcal{S} = (\mathcal{R} + \mathcal{I})^{-1}$ of $\mathcal{R}$ is firmly non-expansive with full domain, it is given by

$$\begin{aligned} \mathcal{S} &= \mathcal{J}_{\lambda T} \circ (2\mathcal{J}_{\boldsymbol{\lambda} U} - \mathcal{I}) + (\mathcal{I} - \mathcal{J}_{\boldsymbol{\lambda} U}) \\ &= \{(\boldsymbol{\lambda} + \rho\boldsymbol{v}, \boldsymbol{\nu} + \rho\boldsymbol{v})\,;\ (\boldsymbol{\nu}, \boldsymbol{\alpha}) \in T, (\boldsymbol{\lambda}, \boldsymbol{v}) \in U \text{ and } \boldsymbol{\nu} + \rho\boldsymbol{\alpha} = \boldsymbol{\lambda} - \rho\boldsymbol{v}\}, \end{aligned} \tag{5.11}$$

with $\text{Fix}\,(\mathcal{S}) = \{\boldsymbol{\lambda} + \rho\boldsymbol{v}; (\boldsymbol{\lambda}, \boldsymbol{v}) \in U, (\boldsymbol{\lambda}, -\boldsymbol{v}) \in T\}$.

Instead of computing $\text{Zer}\,(T + U)$, we can calculate $\bar{\boldsymbol{\zeta}} = \text{Zer}\,(\mathcal{R})$ from which we extract $\text{Zer}\,(T + U) = \mathcal{J}_{\rho U}(\bar{\boldsymbol{\zeta}})$ as shown in the following Lemma.

Lemma 4. *[90] If $\bar{\boldsymbol{\zeta}} = Zer(\mathcal{R})$, then $\bar{\boldsymbol{\lambda}} = \mathcal{J}_{\rho U}(\bar{\boldsymbol{\zeta}}) = Zer\,(T+U)$, where $\mathcal{J}_{\rho U}$ is the resolvent of U given by $\mathcal{J}_{\rho U} = \{(\boldsymbol{\lambda} + \rho\boldsymbol{v}, \boldsymbol{\lambda})\,;\ (\boldsymbol{\lambda}, \boldsymbol{v}) \in U\}$.*

Proof: Let $\bar{\boldsymbol{\zeta}} = \text{Zer}\,(\mathcal{R})$ then, $\bar{\boldsymbol{\zeta}} = \text{Fix}\,(\mathcal{S})$. From the definition of $\text{Fix}\,(\mathcal{S})$, there is a unique couple $(\bar{\boldsymbol{\lambda}}, \bar{\boldsymbol{v}}) \in U$ verifying $\bar{\boldsymbol{\zeta}} = \bar{\boldsymbol{\lambda}} + \rho\,\bar{\boldsymbol{v}}$ and $T(\bar{\boldsymbol{\lambda}}) = -\bar{\boldsymbol{v}}$. Then, from the definition of the resolvent of U we obtain $\mathcal{J}_{\rho U}(\bar{\boldsymbol{\zeta}}) = \mathcal{J}_{\rho U}(\bar{\boldsymbol{\lambda}} + \rho\,\bar{\boldsymbol{v}}) = \bar{\boldsymbol{\lambda}}$ and $(T+U)(\bar{\boldsymbol{\lambda}}) = T(\bar{\boldsymbol{\lambda}}) + U(\bar{\boldsymbol{\lambda}}) = 0$. Thus, $\bar{\boldsymbol{\lambda}} = \text{Zer}\,(T+U)$.

The DR splitting operator $\mathcal{R}$ is a maximal monotone operator hence, we can apply PPA on $\mathcal{R}$. This is easier than applying PPA directly on $D = T + U$. Thus, we will recursively search for $\bar{\boldsymbol{\zeta}} = \text{Fix}\,(\mathcal{S})$.

In the next section we demonstrate how this calculation leads to the solution of our original optimization problem.

5.6 Distributed ADMM

5.6.1 PPA and DR splitting method combined

Using PPA on the DR splitting operator, we establish in this section the distributive computation of the minimum of the convex optimization problem (5.2). Associating PPA with the splitting operator $\mathcal{R}$ leads to the ADMM.

At this point and by applying the PPA, we aim to iteratively calculate $\text{Zer}\,(\mathcal{R})$. In order to develop this iterative process explicitly, we make use of the following Lemma.

Lemma 5. *[68] For any $\boldsymbol{\zeta} = \boldsymbol{\lambda} + \rho\boldsymbol{v}$, such that $(\boldsymbol{\lambda}, \boldsymbol{v}) \in U$ and $\boldsymbol{\lambda} = \mathcal{J}_{\rho U}(\boldsymbol{\zeta})$, there is a variable $\boldsymbol{u}$ such that the following is valid*

i) $\mathcal{S}(\boldsymbol{\zeta}) = \boldsymbol{\lambda} + \rho\tilde{\mathbf{A}}\boldsymbol{u}$,

ii) $\boldsymbol{u} = argmin_{\boldsymbol{u}}\ \mathcal{L}_\rho(\boldsymbol{u}, \boldsymbol{v}; \boldsymbol{\lambda})$,

where, $\rho > 0$ is a penalty parameter and $\mathcal{L}_\rho(\boldsymbol{u}, \boldsymbol{v}; \boldsymbol{\lambda})$ is the augmented Lagrangian of the general problem (5.2) given by

$$L_\rho(\boldsymbol{u}, \boldsymbol{v}; \boldsymbol{\lambda}) \triangleq F(\boldsymbol{u}) + C(\boldsymbol{v}) + \left\langle \boldsymbol{\lambda}, \tilde{\mathbf{A}}\boldsymbol{u} - \boldsymbol{v} \right\rangle + \frac{\rho}{2}\left\| \tilde{\mathbf{A}}\boldsymbol{u} - \boldsymbol{v} \right\|^2. \tag{5.12}$$

Proof: Let $\boldsymbol{\zeta} = \boldsymbol{\lambda} + \rho\boldsymbol{v}$, such that $(\boldsymbol{\lambda}, \boldsymbol{v}) \in U$ and $\boldsymbol{\lambda} = \mathcal{J}_{\rho U}(\boldsymbol{\zeta})$.

i) $\mathcal{R}(\boldsymbol{\zeta}) = \boldsymbol{\lambda} - \boldsymbol{\nu}$ where $(\boldsymbol{\nu}, \boldsymbol{\alpha}) \in T$ and $\boldsymbol{\nu} + \rho\boldsymbol{\alpha} = \boldsymbol{\lambda} - \rho\boldsymbol{v}$. T is the maximal monotone operator given by $T = -\tilde{\mathbf{A}}\partial F^* \circ (-\tilde{\mathbf{A}}^T)$. Therefore,

$$\boldsymbol{\alpha} \in -\tilde{\mathbf{A}}\partial F^*(-\tilde{\mathbf{A}}^T\boldsymbol{\nu})$$

and there is a unique

$$\boldsymbol{u} \in \partial F^*(-\tilde{\mathbf{A}}^T\boldsymbol{\nu})$$

such that

$$\boldsymbol{\alpha} = -\tilde{\mathbf{A}}\boldsymbol{u}.$$

From (5.11) we have

$$\mathcal{S}(\boldsymbol{\zeta}) = \boldsymbol{\nu} + \rho\boldsymbol{v} = \boldsymbol{\lambda} - \rho\boldsymbol{\alpha}.$$

But the following equality holds

$$\boldsymbol{\alpha} = -\tilde{\mathbf{A}}\boldsymbol{u}.$$

Thus, we conclude that

$$\mathcal{S}(\boldsymbol{\zeta}) = \boldsymbol{\lambda} + \rho\tilde{\mathbf{A}}\boldsymbol{u}.$$

ii) Since F is a closed proper convex function, then by the Fenchel-Young inequality [86, Prop 16.9] the following expression

$$\boldsymbol{u} \in \partial F^*(-\tilde{\mathbf{A}}^T\boldsymbol{\nu})$$

is equivalent to

$$-\tilde{\mathbf{A}}^T\boldsymbol{\nu} \in \partial F(\boldsymbol{u}).$$

It follows that

$$0 \in \partial F(\boldsymbol{u}) + \tilde{\mathbf{A}}^T\boldsymbol{\nu}.$$

From the output of (5.10), we have

$$\boldsymbol{\nu} = \boldsymbol{\lambda} - \rho(\boldsymbol{v} + \boldsymbol{\alpha}).$$

Additionally, $\boldsymbol{\alpha}$ can be obtained using

$$\boldsymbol{\alpha} = -\tilde{\mathbf{A}}\boldsymbol{u},$$

then, we obtain:

$$\boldsymbol{\nu} = \boldsymbol{\lambda} + \rho(\tilde{\mathbf{A}}\boldsymbol{u} - \boldsymbol{v}).$$

Subsequently, the expression

$$0 \in \partial F(\boldsymbol{u}) + \tilde{\mathbf{A}}^T\boldsymbol{\nu}$$

translates to

$$0 \in \partial F(\boldsymbol{u}) + \tilde{\mathbf{A}}^T\boldsymbol{\lambda} + \rho\tilde{\mathbf{A}}^T(\tilde{\mathbf{A}}\boldsymbol{u} - \boldsymbol{v}).$$

Using these results, we can conclude that

$$\boldsymbol{u} = \operatorname{argmin}_{\boldsymbol{u}} \mathcal{L}_\rho(\boldsymbol{u}, \boldsymbol{v}; \boldsymbol{\lambda}).$$

Using Lemma $3-5$, we can give explicitly the k^{th} recursion of the PPA applied to $\mathcal{R}$, i.e., the iteration $\boldsymbol{\zeta}^{k+1} = \mathcal{S}(\boldsymbol{\zeta}^k)$. From this iteration we extract the update expressions of $\boldsymbol{u}^k$, $\boldsymbol{v}^k$ and $\boldsymbol{\lambda}^k$ as proven next.

5.6.2 Area-based ADMM

The graph G s decomposed into L overlapping areas A_l, $l \in \{1, \ldots, L\}$. For each area we assign a unique sub-problem where it seeks to update the corresponding variables while having limited coordination with the surrounding areas.

Each area A_l has a subset of vertices $V_l \subset V$ and a subset of edges $E_l = \{\{u, v\}; (u, v) \in V_l^2\} \cap E$ such that the following assumptions hold true.

Assumption 2. *For any $l \in \{1, ..., L\}$, let $G(V_\ell) \triangleq (V_\ell, E_\ell)$ be the sub-graph of area A_ℓ, the following properties are assumed:*

1. $\bigcup_{l=1}^{L} V_\ell = V$,
2. $\bigcup_{l=1}^{L} G(V_\ell)$ *is connected.*

For this decomposition to be clear, we use the following operator:

Definition 1. *We define the linear operator $\prod_{V_l}(\boldsymbol{v})$*

$$\prod_{V_l} : \mathbb{R}^{|v|} \longrightarrow \mathbb{R}^{n_l}$$

$$\boldsymbol{v} \longmapsto (v_\beta)_{(i,j)\in I_l} \tag{5.13}$$

where, $\beta = 2(i-1)N + 2(i-1)m + j$, $I_l = \left\{(i, j); A_{ij} \neq 0, j \in \bigcup_{j\in V_l} J_j\right\}$, $J_j = \{j, j+N, j+2N, j+2N+m\}$, and $n_l = |I_l|$.

Additionally, let $\tilde{\boldsymbol{v}} = [\boldsymbol{v}_1; \ldots; \boldsymbol{v}_L]$, $\tilde{\boldsymbol{\lambda}} = [\boldsymbol{\lambda}_1; \ldots; \boldsymbol{\lambda}_L]$ where $\boldsymbol{v}_l = \prod_{V_l}(\boldsymbol{v})$ and $\boldsymbol{\lambda}_l = \prod_{V_l}(\boldsymbol{\lambda})$ respectively.

We adopt the area-based formulation of the problem given below:[5]

$$\begin{aligned} \min_{\boldsymbol{u}} \quad & F(\boldsymbol{u}) + C(\tilde{\boldsymbol{v}}) \\ \text{subject to} \quad & \tilde{\mathbf{B}}\boldsymbol{u} = \tilde{\boldsymbol{v}} \end{aligned} \tag{5.14}$$

where, C is always defined as an indicator function on the values of $\boldsymbol{v}$, $\tilde{\mathbf{B}}$ is the diagonal map of $\mathbf{B}$. This latter is obtained after projecting $\mathbf{A}$ on area basis using the projection operator defined previously. Thus,

$$\mathbf{A} \underset{\prod}{\longrightarrow} \mathbf{B} = [\mathbf{A_1}; \ldots; \mathbf{A_L}] \underset{\text{diag}}{\longrightarrow} \tilde{\mathbf{B}} = [\text{diag}\,(\mathbf{B})].$$

[5] Depending on the size of the areas, this formulation can be also used to depict a centralized scheme when L is chosen equal to 1. In this case we can apply PPA and DR splitting method on order to obtain the centralized ADMM without the need of the new to project any of the vectors/matrices.

$\mathbf{A_l}$ is the subblock of $\mathbf{A}$ that corresponds to area A_l. It should be noted that size of $\mathbf{B}$ and $\tilde{\mathbf{B}}$ depends on the number of overlapping areas and the number of shared nodes.

At this point, we only substitute $\tilde{\mathbf{A}}, \boldsymbol{v}$ and $\boldsymbol{\lambda}$ by $\tilde{\mathbf{B}}, \tilde{\boldsymbol{v}}$ and $\tilde{\boldsymbol{\lambda}}$ respectively. PPA and DR splitting methods remain applicable.

Using this decomposition, the area-based distributed ADMM algorithm is obtained by the following Lemma.

Lemma 6.

i) Let $\boldsymbol{\zeta}^0 = \tilde{\boldsymbol{\lambda}}^0 + \rho\tilde{\boldsymbol{v}}^0$ *such that* $\tilde{\boldsymbol{\lambda}}^0 = \mathcal{J}_{\rho U}(\boldsymbol{\zeta}^0)$ *and* $(\tilde{\boldsymbol{v}}^0, \tilde{\boldsymbol{\lambda}}^0) \in U$. *Define* $\forall\, k \geq 0$, $\boldsymbol{\zeta}^{k+1} = \mathcal{S}(\boldsymbol{\zeta}^k)$. *Let* $\tilde{\boldsymbol{\lambda}}^k = \mathcal{J}_{\rho U}(\boldsymbol{\zeta}^k)$ *where* $(\tilde{\boldsymbol{v}}^k, \tilde{\boldsymbol{\lambda}}^k) \in U$, *and* $\boldsymbol{u}^{k+1}$ *was uniquely defined by Lemma* 5 *such that* $\mathcal{S}(\boldsymbol{\zeta}^k) = \tilde{\boldsymbol{\lambda}}^k + \rho\tilde{\mathbf{B}}\boldsymbol{u}$[6]. *Then the following holds:*

$$\begin{aligned}
\boldsymbol{u}^{k+1} &= \underset{\boldsymbol{u}}{argmin}\ \mathcal{L}_\rho(\boldsymbol{u}, \tilde{\boldsymbol{v}}^k; \tilde{\boldsymbol{\lambda}}^k),\\
\tilde{\boldsymbol{v}}^{k+1} &= \underset{\tilde{\boldsymbol{v}}}{argmin}\ \mathcal{L}_\rho(\boldsymbol{u}^{k+1}, \tilde{\boldsymbol{v}}; \tilde{\boldsymbol{\lambda}}^k),\\
\tilde{\boldsymbol{\lambda}}^{k+1} &= \tilde{\boldsymbol{\lambda}} + \rho(\tilde{\mathbf{B}}\boldsymbol{u}^{k+1} - \tilde{\boldsymbol{v}}^{k+1}).
\end{aligned}$$

ii) Additionally, let $\mathcal{S}_l(\boldsymbol{\zeta}^k) = \boldsymbol{\lambda}_l^k + \prod_{V_l}(\tilde{\mathbf{B}}\boldsymbol{u}^{k+1})$ *be the* l^{th} *sub-block of* $\mathcal{S}(\boldsymbol{\zeta}^k) = [\boldsymbol{\lambda}_1^k + \prod_{V_1}(\tilde{\mathbf{B}}\boldsymbol{u}^{k+1}); \ldots; \boldsymbol{\lambda}_L^k + \prod_{V_L}(\tilde{\mathbf{B}}\boldsymbol{u}^{k+1})]$ *and* $\boldsymbol{u}_l = \prod_{V_l}(\boldsymbol{u})$. *The sub-problem to be solved by the processor of areas* A_l *is given by:*

$$\boldsymbol{u}_l^{k+1} = \underset{\boldsymbol{u}_l}{argmin} \sum_{j\in V_l} f_j(\boldsymbol{u}_j) + \boldsymbol{\lambda}_l^{k^T} \prod_{V_l}(\tilde{\mathbf{B}}\boldsymbol{u}) + \frac{\rho}{2}\| \prod_{V_l}(\tilde{\mathbf{B}}\boldsymbol{u}) - \boldsymbol{v}_l^k\|^2, \tag{5.15}$$

$$\boldsymbol{v}_l^{k+1} = \underset{\boldsymbol{v}_l}{argmin} - \boldsymbol{\lambda}_l^{k^T}\boldsymbol{v}_l + \frac{\rho}{2}\| \prod_{V_l}(\tilde{\mathbf{B}}\boldsymbol{u}^{k+1}) - \boldsymbol{v}_l\|^2, \tag{5.16}$$

$$\boldsymbol{\lambda}_l^{k+1} = \boldsymbol{\lambda}_l^k + \rho(\prod_{V_l}(\tilde{\mathbf{B}}\boldsymbol{u}^{k+1}) - \boldsymbol{v}_l^{k+1}), \tag{5.17}$$

Proof: For the completeness of our work, we provide this proof that follows the work in [68].

i) Let $\boldsymbol{\zeta}^k = \tilde{\boldsymbol{\lambda}}^k + \rho\tilde{\boldsymbol{v}}^k$, by Lemma 3 there is a unique $\boldsymbol{u}^{k+1}$ such that

$$\mathcal{S}(\boldsymbol{\zeta}^k) = \boldsymbol{\nu}^k + \rho\tilde{\boldsymbol{v}}^k = \tilde{\boldsymbol{\lambda}}^k + \rho\tilde{\mathbf{B}}\boldsymbol{u}^{k+1}$$

[6] $\mathbf{B}$ is obtained through the projection of $\mathbf{A}$ to the subset V_l of each area A_l, and $\tilde{\mathbf{B}} = diag\,[\mathbf{B}]$

and

$$\boldsymbol{u}^{k+1} = \text{argmin}_{\boldsymbol{u}}\ \mathcal{L}_\rho(\boldsymbol{u}, \tilde{\boldsymbol{v}}^k; \tilde{\boldsymbol{\lambda}}^k).$$

As for $\boldsymbol{\lambda}^{k+1}$ and $\boldsymbol{v}^{k+1}$, we derive their corresponding expressions using the following hypothesis on $\boldsymbol{\zeta}$:

$$\boldsymbol{\zeta}^{k+1} = \mathcal{S}(\boldsymbol{\zeta}^k).$$

On the one hand, let $\boldsymbol{\zeta}^{k+1} = \tilde{\boldsymbol{\lambda}}^{k+1} + \rho\tilde{\boldsymbol{v}}^{k+1}$ where $(\tilde{\boldsymbol{\lambda}}^{k+1}, \tilde{\boldsymbol{v}}^{k+1}) \in U$ and $\tilde{\boldsymbol{\lambda}}^{k+1} = \mathcal{J}_{\rho U}(\boldsymbol{\zeta}^{k+1})$ then,

$$\tilde{\boldsymbol{\lambda}}^{k+1} = \boldsymbol{\zeta}^{k+1} - \rho\tilde{\boldsymbol{v}}^{k+1}.$$

On the other hand, $\mathcal{S}(\boldsymbol{\zeta}^k) = \tilde{\boldsymbol{\lambda}}^k + \rho\tilde{\mathbf{B}}\boldsymbol{u}^{k+1} = \boldsymbol{\zeta}^{k+1}$. Thus,

$$\tilde{\boldsymbol{\lambda}}^{k+1} = \tilde{\boldsymbol{\lambda}}^k + \rho\tilde{\mathbf{B}}\boldsymbol{u}^{k+1} - \rho\tilde{\boldsymbol{v}}^{k+1}.$$

Moreover

$$\tilde{\boldsymbol{v}}^{k+1} \in U(\tilde{\boldsymbol{\lambda}}^{k+1})$$

then, by the Fenchel-Young inequality [86, Prop 16.9] this is equivalent to

$$\tilde{\boldsymbol{\lambda}}^{k+1} \in \partial C(\tilde{\boldsymbol{v}}^{k+1}).$$

Thus, we have

$$0 \in \partial C(\tilde{\boldsymbol{v}}^{k+1}) - \tilde{\boldsymbol{\lambda}}^{k+1}.$$

It follows that

$$0 \in \partial C(\tilde{\boldsymbol{v}}^{k+1}) - \tilde{\boldsymbol{\lambda}}^k + \rho\tilde{\boldsymbol{v}}^{k+1} - \rho\tilde{\mathbf{B}}\boldsymbol{u}^{k+1},$$

which is equivalent to

$$\tilde{\boldsymbol{v}}^{k+1} = \text{argmin}_{\tilde{\boldsymbol{v}}}\ \mathcal{L}_\rho(\boldsymbol{u}^{k+1}, \tilde{\boldsymbol{v}}; \tilde{\boldsymbol{\lambda}}^k).$$

We conclude that there is a unique $\boldsymbol{u}^{k+1}$ such that $\mathcal{S}(\boldsymbol{\zeta}^k) = \boldsymbol{\nu}^k + \rho\tilde{\boldsymbol{v}}^k = \tilde{\boldsymbol{\lambda}}^k + \rho\tilde{\mathbf{B}}\boldsymbol{u}^{k+1}$ where,

$$\boldsymbol{u}^{k+1} = \text{argmin}_{\boldsymbol{u}}\ \mathcal{L}_\rho(\boldsymbol{u}, \tilde{\boldsymbol{v}}^k; \tilde{\boldsymbol{\lambda}}^k) \tag{5.18}$$

$$\tilde{\boldsymbol{v}}^{k+1} = \text{argmin}_{\tilde{\boldsymbol{v}}}\ \mathcal{L}_\rho(\boldsymbol{u}^{k+1}, \tilde{\boldsymbol{v}}; \tilde{\boldsymbol{\lambda}}^k) \tag{5.19}$$

$$\tilde{\boldsymbol{\lambda}}^{k+1} = \tilde{\boldsymbol{\lambda}}^k + \rho(\tilde{\mathbf{B}}\boldsymbol{u}^{k+1} - \tilde{\boldsymbol{v}}^{k+1}). \tag{5.20}$$

ii) Hereafter, we show that the decomposition stated in this Lemma is true for the update expression of $\boldsymbol{u}$.

$$\begin{aligned}\boldsymbol{u}^{k+1} &= \underset{\boldsymbol{u}}{\operatorname{argmin}}\, F(\boldsymbol{u}) + C(\tilde{\boldsymbol{v}}^k) + \langle \tilde{\boldsymbol{\lambda}}^k, \tilde{\mathbf{B}}\boldsymbol{u} - \tilde{\boldsymbol{v}}^k\rangle + \frac{\rho}{2}\|\tilde{\mathbf{B}}\boldsymbol{u} - \tilde{\boldsymbol{v}}^k\|^2 \\ &= \underset{\boldsymbol{u}}{\operatorname{argmin}}\, F(\boldsymbol{u}) + \sum_{l=1}^{L}\left\{ \boldsymbol{\lambda}_l^{k^T} \prod_{V_l}(\tilde{\mathbf{B}}\boldsymbol{u}) + \frac{\rho}{2}\| \prod_{V_l}(\tilde{\mathbf{B}}\boldsymbol{u}) - \boldsymbol{v}_l^k\|^2 \right\}.\end{aligned} \tag{5.21}$$

This expression is separable into L independent parts $\boldsymbol{u}_l^{k+1}, l \in \{1, \ldots, L\}$. Each is given by $\boldsymbol{u}_l = \prod_{V_l}(\boldsymbol{u})$.

The sub-block $\boldsymbol{u}_l^{k+1}$ is assigned to A_l and contains only the components of $\boldsymbol{u}^{k+1}$ corresponding to the nodes $j \in V_l$.

$$\boldsymbol{u}_l^{k+1} = \underset{\boldsymbol{u}_l}{\operatorname{argmin}} \sum_{j\in V_l} f_j(u_j) + \boldsymbol{\lambda}_l^{k^T} \prod_{V_l}(\tilde{\mathbf{B}}\boldsymbol{u}) + \frac{\rho}{2}\| \prod_{V_l}(\tilde{\mathbf{B}}\boldsymbol{u}) - \boldsymbol{v}_l^k\|^2 \tag{5.22}$$

As for $\tilde{\boldsymbol{v}}$, we have:

$$\begin{aligned}\tilde{\boldsymbol{v}}^{k+1} &= \underset{\tilde{\boldsymbol{v}}}{\operatorname{argmin}}\, F(\boldsymbol{u}^{k+1}) + C(\tilde{\boldsymbol{v}}) + \langle \tilde{\boldsymbol{\lambda}}^k, \tilde{\mathbf{B}}\boldsymbol{u}^{k+1} - \tilde{\boldsymbol{v}}\rangle + \frac{\rho}{2}\|\tilde{\mathbf{B}}\boldsymbol{u}^{k+1} - \tilde{\boldsymbol{v}}\|^2 \\ &= \underset{\tilde{\boldsymbol{v}}}{\operatorname{argmin}}\, C(\tilde{\boldsymbol{v}}) + \sum_{l=1}^{L}\left\{ -\boldsymbol{\lambda}_l^{k^T}\boldsymbol{v}_l + \frac{\rho}{2}\| \prod_{V_l}(\tilde{\mathbf{B}}\boldsymbol{u}^{k+1}) - \boldsymbol{v}_l\|^2 \right\}.\end{aligned} \tag{5.23}$$

Thus, each area has to solve:

$$\boldsymbol{v}_l^{k+1} = \underset{\boldsymbol{v}_l}{\operatorname{argmin}}\, C(\tilde{\boldsymbol{v}}) - \boldsymbol{\lambda}_l^{k^T}\boldsymbol{v}_l + \frac{\rho}{2}\| \prod_{V_l}(\tilde{\mathbf{B}}\boldsymbol{u}^{k+1}) - \boldsymbol{v}_l\|^2 \tag{5.24}$$

and,

$$\boldsymbol{\lambda}_l^{k+1} = \boldsymbol{\lambda}_l^k + \rho(\prod_{V_l}(\tilde{\mathbf{B}}\boldsymbol{u}^{k+1}) - \boldsymbol{v}_l^{k+1}). \tag{5.25}$$

At this point, we were able to demonstrate how efficiently we can obtain the area-based ADMM equations using PPA and DR splitting method.

5.6.3 Classical ADMM

If we did not decompose the graph into overlapping areas then, the direct application of the PPA and DR splitting method would have led to the classical ADMM algorithm.

This can be also seen as a special case of the area based distributed ADMM algorithm where we only have one area of size N is considered.

An iteration of the classical ADMM algorithm that we obtain is given by:

$$\boldsymbol{u}^{k+1} = \underset{\boldsymbol{u}}{\operatorname{argmin}} \left\{ F(\boldsymbol{u}) + \frac{\rho}{2} \left\| \tilde{\mathbf{A}}\boldsymbol{u} - \boldsymbol{v}^k + \frac{\boldsymbol{\lambda}^k}{\rho} \right\|^2 \right\} \tag{5.26a}$$

$$\boldsymbol{v}^{k+1} = \underset{\boldsymbol{v}}{\operatorname{argmin}} \left\{ C(\boldsymbol{v}) + \frac{\rho}{2} \left\| \tilde{\mathbf{A}}\boldsymbol{u}^{k+1} - \boldsymbol{v} + \frac{\boldsymbol{\lambda}^k}{\rho} \right\|^2 \right\} \tag{5.26b}$$

$$\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + \rho \left(\tilde{\mathbf{A}}\boldsymbol{u}^{k+1} - \boldsymbol{v}^{k+1} \right) \tag{5.26c}$$

5.7 Conclusion

We showed how the ADMM method can be applied to the Production-Sharing problem to make it distributed on area basis. In this context, the problem is divided into different overlapping areas. Each sub-problem is solved by a dedicated processor in the corresponding area. In the next chapter we will provide the main results of our work. We present the classical ADMM application to the DPS problem then, we demonstrate how an asynchronous version of the area based ADMM can be applied effectively on the DPS problem. We also extend these results to a network with non-overlapping areas.

Chapter 6

DPS w/ADMM

6.1 Introduction

We derived in the previous chapters a DPS problem. The DC-OPF problem was shown to be a special case of the DPS problem. We also demonstrated that the DPS problem is well suited for the distributed ADMM. We provide in this chapter the detailed application of the ADMM to the DPS problem, starting from its classical form. Then, we consider the case where the network is divided into multiple overlapping areas and we present the distributed ADMM application. Confronted by the synchronicity of this application, we present an asynchronous application of the ADMM and prove its convergence. Additionally, we extend our results to the case of a network with non-overlapping areas, a case that appears more feasible and realistic.

6.2 Classical ADMM application to the Production-Sharing problem

One can remark that the DPS problem (4.22) is composed of i) a convex function F; ii) and (sum of) indicator function C; and iii) a relation matrix $\tilde{\mathbf{A}}$. While there is not a lot of assumptions on function F, function C and matrix $\tilde{\mathbf{A}}$ have a very specific structure.

Indeed, C is an indicator function of certain structure in its argument and $\tilde{\mathbf{A}}$ is a network related matrix; their composition leads to a function ensuring that the optimization variable verifies the original constraints of Problem (4.1).

As explained in the previous chapter, it is interesting to break the composition $C \circ \tilde{\mathbf{A}}$; and split the resolution of the problem between F and C.

To do so, it is common to consider the dual of the problem, we also split the use of sub-gradient of the dual function using DR splitting [68]. This conjunction along with PPA actually leads to the well-known ADMM.

Writing an iteration of the ADMM algorithm on Problem (4.22) leads to the following set of equations

$$\boldsymbol{u}^{k+1} = \underset{\boldsymbol{u}}{\operatorname{argmin}} \left\{ F(\boldsymbol{u}) + \frac{\rho}{2} \left\| \tilde{\mathbf{A}}\boldsymbol{u} - \boldsymbol{v}^k + \frac{\boldsymbol{\lambda}^k}{\rho} \right\|^2 \right\} \tag{6.1a}$$

$$\boldsymbol{v}^{k+1} = \underset{\boldsymbol{v}}{\operatorname{argmin}} \left\{ C(\boldsymbol{v}) + \frac{\rho}{2} \left\| \tilde{\mathbf{A}}\boldsymbol{u}^{k+1} - \boldsymbol{v} + \frac{\boldsymbol{\lambda}^k}{\rho} \right\|^2 \right\} \tag{6.1b}$$

$$\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + \rho \left(\tilde{\mathbf{A}}\boldsymbol{u}^{k+1} - \boldsymbol{v}^{k+1} \right) \tag{6.1c}$$

where, $\rho > 0$ is a free hyper-parameter and $\boldsymbol{\lambda}$ is the vector of Lagrangian multipliers.

We decompose these equations as follows.

For every node $j \in V$, let $I_j = \{i : A_{ip} \neq 0, p = \{j, j+N, j+2N, j+2N+m\}\}$ be the set of constraints in which node j is involved. The minimization step (6.1a) can be written as follows:

$$\begin{aligned}
\boldsymbol{u}^{k+1} = \underset{\substack{x_j, z_j, \\ j=1,\dots,N}}{\operatorname{argmin}} \; & I_+(\mathbf{s_1}) + I_+(\mathbf{s_2}) + \sum_{j=1}^{N} f_j(x_j) \\
& + \sum_{i=1}^{N+2m} \frac{\rho}{2} \Bigg\{ \sum_{j=1}^{N} \Bigg\{ \left\| A_{ij} x_j + \frac{\lambda^k_\beta}{\rho} - v^k_\beta \right\|^2 \\
& + \left\| A_{ij_{+N}} z_j + \frac{\lambda^k_{\beta+N}}{\rho} - v^k_{\beta+N} \right\|^2 \Bigg\} \\
& + \sum_{j=1}^{m} \Bigg\{ \left\| A_{ij_{+2N}} s_{1,j} + \frac{\lambda^k_{\beta+2N}}{\rho} - v^k_{\beta+2N} \right\|^2 \\
& + \left\| A_{ij_{+2N+m}} s_{2,j} + \frac{\lambda^k_{\beta+2N+m}}{\rho} - v^k_{\beta+2N+m} \right\|^2 \Bigg\} \Bigg\}
\end{aligned} \tag{6.2}$$

where, $\beta = \beta(i,j) = 2N(i-1) + 2m(i-1) + j$ is used as a compact notation to index the values of $\boldsymbol{v}$ and $\boldsymbol{\lambda}$ corresponding to the element A_{ij}. Similarly $\beta + N$, $\beta + 2N$ and $\beta + 2N + m$ correspond to $A_{ij_{+N}}$, $A_{ij_{+2N}}$ and $A_{ij_{+2N+m}}$ respectively.

This expression is separable into node basis because the expressions used to update x_j and z_j depend only on the previous values calculated for node

j and its one-hop neighbors. Thus the iteration on $\boldsymbol{u}$ can be implemented in a distributed manner where each node $j \in V$ update its variables x_j, z_j by solving the following update steps.[1]

$$x_j^{k+1} = \underset{x_j}{\text{argmin}}\ f_j(x_j) + \frac{\rho}{2}\sum_{i=1}^{N+2m} \|A_{ij}x_j + \frac{\lambda_\beta^k}{\rho} - v_\beta^k\|^2 \tag{6.3}$$

$$z_j^{k+1} = \underset{z_j}{\text{argmin}}\ \frac{\rho}{2}\sum_{i=1}^{N+2m} \|A_{ij_{+N}}z_j + \frac{\lambda_{\beta+N}^k}{\rho} - v_{\beta+N}^k\|^2 \tag{6.4}$$

$$s_{1,j}^{k+1} = \underset{s_{1,j}}{\text{argmin}}\ \frac{\rho}{2}\sum_{i=1}^{N+2m} \|A_{ij_{+2N}}s_{1,j} + \frac{\lambda_{\beta+2N}^k}{\rho} - v_{\beta+2N}^k\|^2 \tag{6.5}$$

$$s_{2,j}^{k+1} = \underset{s_{2,j}}{\text{argmin}}\ \frac{\rho}{2}\sum_{i=1}^{N+2m} \|A_{ij_{+2N+m}}s_{2,j} + \frac{\lambda_{\beta+2N+m}^k}{\rho} - v_{\beta+2N+m}^k\|^2 \tag{6.6}$$

These update steps can be further simplified using the specific structure of matrix $\mathbf{A}$ and compact notations $\alpha = \beta(j,j) = (2N+2m+1)(j-1)+1$, $\gamma = \beta(i+N,j) = 2N(i+N-1)+2m(i+N-1)+j$, $\omega = \beta(i+N+m,j) = 2N(i+N+m-1)+2m(i+N+m-1)+j$, $\kappa = \beta(i,i-N) = 2N(i-1)+2m(i-1)+i-N$, and $\chi = \beta(i,i-N-m) = 2N(i-1)+2m(i-1)+i-N-m$.

First we need to explicit the update step on the auxiliary variables (6.1b):

$$\begin{aligned}
\boldsymbol{v}^{k+1} = \underset{\boldsymbol{v}}{\text{argmin}}\ C(\boldsymbol{v}) + \frac{\rho}{2}\sum_{i=1}^{N+2m}\Bigg\{ & \sum_{j=1}^{N}\Bigg\{ \left\| A_{ij}x_j^{k+1} - v_\beta + \frac{{\lambda_\beta}^k}{\rho}\right\|^2 \\
& + \left\| A_{ij_{+N}}z_j^{k+1} - v_{\beta_{+N}} + \frac{{\lambda_{\beta_{+N}}}^k}{\rho}\right\|^2 \Bigg\} \\
& + \sum_{j=1}^{m}\Bigg\{ \left\| A_{ij_{+2N}}s_{1,j}^{k+1} - v_{\beta_{+2N}} + \frac{{\lambda_{\beta_{+2N}}}^k}{\rho}\right\|^2 \\
& + \left\| A_{ij_{+2N+m}}s_{1,j}^{k+1} - v_{\beta_{+2N+m}} + \frac{{\lambda_{\beta_{+2N+m}}}^k}{\rho}\right\|^2 \Bigg\}\Bigg\} \\
\text{subject to } & \sum_{j=1}^{2N+2m} v_\beta = \tilde{d}_i, \forall i = 1,\dots,N+2m.
\end{aligned} \tag{6.7}$$

This expression can be divided into a set of four equations using the

[1] The indicator on the sign of $s_{1,j}$ and $s_{2,j}$ will appear later as a projection of these variables into the set of positive numbers.

compact notation β as follows:

$$v_{\beta}^{k+1} = \underset{v_{\beta}}{\operatorname{argmin}} \frac{\rho}{2} \left\| A_{ij} x_j^{k+1} - v_{\beta} + \frac{{\lambda_{\beta}}^k}{\rho} \right\|^2 \tag{6.8a}$$

$$v_{\beta_{+N}}^{k+1} = \underset{v_{\beta_{+N}}}{\operatorname{argmin}} \frac{\rho}{2} \left\| A_{ij_{+N}} z_j^{k+1} - v_{\beta_{+N}} + \frac{{\lambda_{\beta_{+N}}}^k}{\rho} \right\|^2 \tag{6.8b}$$

$$v_{\beta_{+2N}}^{k+1} = \underset{v_{\beta_{+2N}}}{\operatorname{argmin}} \frac{\rho}{2} \left\| A_{ij_{+2N}} s_{1,j}^{k+1} - v_{\beta_{+2N}} + \frac{{\lambda_{\beta_{+2N}}}^k}{\rho} \right\|^2 \tag{6.8c}$$

$$v_{\beta_{+2N+m}}^{k+1} = \underset{v_{\beta_{+2N}}}{\operatorname{argmin}} \frac{\rho}{2} \left\| A_{ij_{+2N+m}} s_{2,j}^{k+1} - v_{\beta_{+2N+m}} + \frac{{\lambda_{\beta_{+2N+m}}}^k}{\rho} \right\|^2 \tag{6.8d}$$

$$\text{subject to} \sum_{j=1}^{N} (v_{\beta} + v_{\beta_{+N}}) + \sum_{j=1}^{m} (v_{\beta_{+2N}} + v_{\beta_{+2N+m}}) = \tilde{d}_i, \forall i = 1, \ldots, N + 2m.$$

In order to solve this set of minimization steps we introduce the vector of Lagrangian multipliers $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_{N+2m})^T$ to the set of constraints $\tilde{\mathbf{A}}\boldsymbol{u} = \tilde{\mathbf{d}}$. Thus, $\boldsymbol{\pi}$ correspond to the set of constraints on the entries of $\boldsymbol{v}$. After some algebra we obtain:

$$v_{\beta}^{k+1} = \frac{1}{\rho}(-\pi_i^{k+1} + \lambda_{\beta}^k) + A_{ij} x_j^{k+1} \tag{6.9a}$$

$$v_{\beta_{+N}}^{k+1} = \frac{1}{\rho}(-\pi_i^{k+1} + \lambda_{\beta_{+N}}^k) + A_{ij_{+N}} z_j^{k+1} \tag{6.9b}$$

$$v_{\beta_{+2N}}^{k+1} = \frac{1}{\rho}(-\pi_i^{k+1} + \lambda_{\beta_{+2N}}^k) + A_{ij_{+2N}} s_{1,j}^{k+1} \tag{6.9c}$$

$$v_{\beta_{+2N+m}}^{k+1} = \frac{1}{\rho}(-\pi_i^{k+1} + \lambda_{\beta_{+2N+m}}^k) + A_{ij_{+2N+m}} s_{2,j}^{k+1}. \tag{6.9d}$$

We substitute these results into their corresponding constraints which gives us:

$$\pi_i^{k+1} = \frac{1}{d(i)} \Big\{ \rho r_i(\boldsymbol{u}^{k+1}) + \sum_{j=1}^{N} (\lambda_{\beta}^k + \lambda_{\beta_{+N}}^k) + \sum_{j=1}^{m} (\lambda_{\beta_{+2N}}^k + \lambda_{\beta_{+2N+m}}^k) \Big\} \tag{6.10}$$

where $d(i)$ is the degree of the i^{th} constraint (i.e., the count of its nonzero

elements), and $r_i^k \hat{=} [\mathbf{A}]_i \mathbf{u}^k - \tilde{d}_i$. The residual of the i^{th} constraint is given by:

$$\begin{aligned} r_i^k = &-d_i + \mathbf{1}_{i\leq N}(\sum_{j=1}^{N} x_j^k + L_{ij} z_j^k) \\ &+ \mathbf{1}_{N<i\leq N+m}(\sum_{j=1}^{N} M_{ij} z_j^k + \sum_{j=1}^{m} s_{1,j}^k) \\ &+ \mathbf{1}_{i>N+m}(\sum_{j=1}^{N} - M_{ij} z_j^k + \sum_{j=1}^{m} s_{2,j}^k) \end{aligned} \tag{6.11}$$

where $\mathbf{1}_{i<N} = 1$ when $i < N$ and 0 otherwise, vice versa for $\mathbf{1}_{i>N}$.

As for the dual variable update step (6.1c), we can substitute it with a set of separated updates of its components λ_β. Then, by making use of (6.9a) we can reduce the computational complexity of the dual update step (6.1c) by replacing it with a set of updates on π. This is proved as follows,

$$\begin{aligned} &\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + \rho\left(\tilde{\mathbf{A}}\boldsymbol{u}^{k+1} - \boldsymbol{v}^{k+1}\right) \\ &\Rightarrow \lambda_\beta^{k+1} = \lambda_\beta^k + \rho(A_{ij} x_j^{k+1} - v_\beta^{k+1}) && \text{(6.12a)} \\ &\quad \lambda_{\beta_{+N}}^{k+1} = \lambda_{\beta_{+N}}^k + \rho(A_{ij_{+N}} z_j^{k+1} - v_{\beta_{+N}}^{k+1}) && \text{(6.12b)} \\ &\quad \lambda_{\beta_{+2N}}^{k+1} = \lambda_{\beta_{+2N}}^k + \rho(A_{ij_{+2N}} s_{1,j}^{k+1} - v_{\beta_{+2N}}^{k+1}) && \text{(6.12c)} \\ &\quad \lambda_{\beta_{+2N+m}}^{k+1} = \lambda_{\beta_{+2N+m}}^k + \rho(A_{ij_{+2N+m}} s_{2,j}^{k+1} - v_{\beta_{+2N+m}}^{k+1}). && \text{(6.12d)} \end{aligned}$$

Equation (6.12a) yields

$$\begin{aligned} \lambda_\beta^{k+1} =& \lambda_\beta^k + \rho(A_{ij} x_j^{k+1} - \frac{1}{\rho}(-\pi_i^{k+1} + \lambda_\beta^k) - A_{ij} x_j^{k+1}) \\ =& \pi_i^{k+1}, \quad \forall j = 1, \ldots, N. \end{aligned} \tag{6.13}$$

The same equality holds for the other entries of $\boldsymbol{\lambda}$ i.e., for each constraint $i = 1, \ldots, N + 2m$ we have the equality between every corresponding entry in $\boldsymbol{\lambda}$ and the corresponding π_i. That is, $\lambda_\beta^{k+1} = \pi_i^{k+1}$, $\forall j = 1, \ldots, 2N + 2m$.

We plug this last result into (6.9a) (6.9b) and (6.9d), we obtain:

$$v_{\beta}^{k+1} = A_{ij} x_j^{k+1} - \frac{r_i^{k+1}}{d(i)} \tag{6.14a}$$

$$v_{\beta_{+N}}^{k+1} = A_{ij_{+N}} z_j^{k+1} - \frac{r_i^{k+1}}{d(i)} \tag{6.14b}$$

$$v_{\beta_{+2N}}^{k+1} = A_{ij_{+2N}} s_{1,j}^{k+1} - \frac{r_i^{k+1}}{d(i)} \tag{6.14c}$$

$$v_{\beta_{+2N+m}} = A_{ij_{+2N+m}} s_{2,j}^{k+1} - \frac{r_i^{k+1}}{d(i)}. \tag{6.14d}$$

Which can be reduced using (6.11) into:

$$v_{\beta}^{k+1} = \mathbf{1}_{i=j}\, x_j^{k+1} - \frac{r_i^{k+1}}{d(i)} \tag{6.15a}$$

$$v_{\beta_{+N}}^{k+1} = \mathbf{1}_{i\leq N} L_{i_{-N}j}\, z_j^{k+1} - \frac{r_i^{k+1}}{d(i)} \tag{6.15b}$$

$$v_{\beta_{+2N}}^{k+1} = \mathbf{1}_{N<i\leq N+m}\, s_{1,j}^{k+1} - \frac{r_i^{k+1}}{d(i)} \tag{6.15c}$$

$$v_{\beta_{+2N+m}} = \mathbf{1}_{i>N+m}\, s_{2,j}^{k+1} - \frac{r_i^{k+1}}{d(i)}. \tag{6.15d}$$

After a careful look at these equations we can deduce that a big numbers of the entries in $\boldsymbol{v}$ are always zeroes, these entries correspond to the zero elements in $\mathbf{A}$. These updates correspond to the difference between the updated primal variables and the corresponding constraints' residuals. Thus, they can be directly taken into account in the other steps of the distributed optimization algorithm.

As for (6.10), it reduces to the following:

$$\pi_i^{k+1} = \frac{\rho}{d(i)} r_i^{k+1} + \pi_i^k. \tag{6.16}$$

We adopt a new formulation for the residual $\boldsymbol{r} = (\boldsymbol{r}_1, \boldsymbol{r}_2, \boldsymbol{r}_2)$ where,

$$r_{1,i}^k \hat{=} r_i^k = \mathbf{1}_{i=j} x_j^k + \sum_{j=1}^{N} L_{ij} z_j^k - d_i, \forall i = 1, \dots, N \tag{6.17}$$

$$r_{2,i}^k \hat{=} r_{i_{+N}}^k = \mathbf{1}_{i=j} s_{1,j}^k + \sum_{j=1}^{N} M_{ij} z_j^k - d_{i+N}, \forall i = 1, \dots, m \tag{6.18}$$

$$r_{3,i}^k \hat{=} r_{i_{+N+m}}^k = \mathbf{1}_{i=j} s_{2,j}^k - \sum_{j=1}^{N} M_{ij} z_j^k - d_{i+N+m}, \forall i = 1, \dots, m. \tag{6.19}$$

In the same manner, we divide $\boldsymbol{\pi}$ into $\boldsymbol{\pi}_1 \in \mathbb{R}^N$, $\boldsymbol{\pi}_2 \in \mathbb{R}^m$ and $\boldsymbol{\pi}_3 \in \mathbb{R}^m$, after some algebra we obtain:

$$\pi_{1,i}^{k+1} \hat{=} \pi_i^{k+1} = \frac{\rho}{d_1(i)} r_{1,i}^{k+1} + \pi_{1,i}^k \tag{6.20}$$

$$\pi_{2,i}^{k+1} \hat{=} \pi_{i_{+N}}^{k+1} = \frac{\rho}{d_2(i)} r_{2,i}^{k+1} + \pi_{2,i}^k \tag{6.21}$$

$$\pi_{3,i}^{k+1} \hat{=} \pi_{i_{+N+m}}^{k+1} = \frac{\rho}{d_3(i)} r_{3,i}^{k+1} + \pi_{3,i}^k \tag{6.22}$$

where, $d_1(i)$, $d_2(i)$ and $d_3(i)$ represent the degree of constraint i, $i+N$ and $i+N+m$ respectively.

Finally, we can rewrite the primal variables update steps (6.3) and (6.4) as:

$$\begin{aligned} x_j^{k+1} &= \underset{x_j}{\text{argmin}}\ f_j(x_j) + \frac{\rho}{2} \sum_{i=1}^{N} \| A_{ij} x_j + \frac{\lambda_\beta^k}{\rho} - v_\beta^k \|^2 + \frac{\rho}{2} \sum_{i=N+1}^{N+2m} \| \frac{\lambda_\beta^k}{\rho} - v_\beta^k \|^2 \\ &= \underset{x_j}{\text{argmin}}\ f_j(x_j) + \frac{\rho}{2} \| x_j + \frac{\lambda_\alpha^k}{\rho} - v_\alpha^k \|^2 \end{aligned} \tag{6.23}$$

$$\begin{aligned} z_j^{k+1} &= \underset{z_j}{\text{argmin}}\ \frac{\rho}{2} \Big\{ \sum_{i=1}^{N} \| L_{ij} z_j + \frac{\lambda_{\beta_{+N}}^k}{\rho} - v_{\beta_{+N}}^k \|^2 + \sum_{i=N+1}^{N+m} \| M_{i_{-N}j} z_j + \frac{\lambda_{\beta_{+N}}^k}{\rho} - v_{\beta_{+N}}^k \|^2 \\ &+ \sum_{i=N+m+1}^{N+2m} \| - M_{i_{-N-m}j} z_j + \frac{\lambda_{\beta_{+N}}^k}{\rho} - v_{\beta_{+N}}^k \|^2 \Big\} \\ &= \underset{z_j}{\text{argmin}}\ \frac{\rho}{2} \sum_{i=1}^{N} \| L_{ij} z_j + \frac{\lambda_{\beta_{+N}}^k}{\rho} - v_{\beta_{+N}}^k \|^2 + \\ &+ \frac{\rho}{2} \sum_{i=1}^{m} \Big\{ \| M_{ij} z_j + \frac{\lambda_{\gamma_{+N}}^k}{\rho} - v_{\gamma_{+N}}^k \|^2 \| - M_{ij} z_j + \frac{\lambda_{\omega_{+N}}^k}{\rho} - v_{\omega_{+N}}^k \|^2 \Big\} \end{aligned} \tag{6.24}$$

In the same manner, we can simplify the slack variables update steps (6.5) and (6.6) as follows:

$$s_{1,j}^{k+1} = \underset{s_{1,j}}{\operatorname{argmin}} \frac{\rho}{2}\|s_{1,j} + \frac{\lambda_{\kappa+2N}^k}{\rho} - v_{\kappa+2N}^k\|^2 \tag{6.25}$$

$$s_{2,j}^{k+1} = \underset{s_{2,j}}{\operatorname{argmin}} \frac{\rho}{2}\|s_{2,j} + \frac{\lambda_{\chi+2N+m}^k}{\rho} - v_{\chi+2N+m}^k\|^2 \tag{6.26}$$

Using the previous results on the auxiliary variables and Lagrangian multipliers, we obtain the following update steps for the primal and slack variables:

$$x_j^{k+1} = \underset{x_j}{\operatorname{argmin}} \left\{ f_j(x_j) + \frac{\rho}{2}\|x_j + \frac{\pi_{1,j}^k}{\rho} - x_j^k + \frac{r_{1,j}^k}{d_1(j)}\|^2 \right\} \tag{6.27}$$

$$\begin{aligned} z_j^{k+1} =& \underset{z_j}{\operatorname{argmin}} \sum_{i=1}^{N} \|L_{ij}z_j + \frac{\pi_{1,i}^k}{\rho} - L_{ij}z_j^k + \frac{r_{1,i}^k}{d_1(i)}\|^2 \\ &+ \sum_{i=1}^{m} \Big\{ \|M_{ij}z_j + \frac{\pi_{2,i}^k}{\rho} - M_{ij}z_j^k + \frac{r_{2,i}^k}{d_2(i)}\|^2 \\ &+ \| - M_{ij}z_j + \frac{\pi_{3,i}^k}{\rho} + M_{ij}z_j^k + \frac{r_{3,i}^k}{d_3(i)}\|^2 \Big\} \end{aligned} \tag{6.28}$$

$$s_{1,j}^{k+1} = \max\left\{0, z_{1,j}^k - \frac{\pi_{2,j}^k}{\rho} - \frac{r_{2,j}^k}{d_2(i)}\right\} \tag{6.29}$$

$$s_{2,j}^{k+1} = \max\left\{0, z_{2,j}^k - \frac{\pi_{3,j}^k}{\rho} - \frac{r_{3,j}^k}{d_3(i)}\right\} \tag{6.30}$$

Solving the update steps leads to Algorithm 1 on Production-Sharing algorithm using ADMM.

Algorithm 1: Production-Sharing optimization w/ ADMM

1. Initialize $\boldsymbol{u}$ and $\boldsymbol{\pi}$ to the initial values $\boldsymbol{u}^0$ and $\boldsymbol{\pi}^0$.

2. At iteration k:

   (a) For every agent $j = 1, \ldots, N$ update x_j and z_j:

$$x_j^{k+1} = \underset{x_j}{\operatorname{argmin}} f_j(x_j) + \frac{\rho}{2}\|x_j + \frac{\pi_{1,j}^k}{\rho} - x_j^k + \frac{r_{1,j}^k}{d_1(j)}\|^2 \tag{6.31a}$$

$$z_j^{k+1} = z_j^k - \frac{1}{\sum_{i=1}^N L_{ij}^2 + 2\sum_{i=1}^m M_{ij}^2}\left(\sum_{i=1}^N L_{ij}\Big(\frac{\pi_{1,i}^k}{\rho} + \frac{r_{1,i}^k}{d_1(i)}\Big) + \sum_{i=1}^m M_{ij}\Big(\frac{\pi_{2,i}^k}{\rho} + \frac{r_{2,i}^k}{d_2(i)}\Big) - \sum_{i=1}^m M_{ij}\Big(\frac{\pi_{3,i}^k}{\rho} + \frac{r_{3,i}^k}{d_3(i)}\Big)\right). \tag{6.31b}$$

   For each connection $j = 1, \ldots, m$, the corresponding nodes update $s_{1,j}$ and $s_{2,j}$ using:

$$s_{1,j}^{k+1} = \max\Big\{0, z_{1,j}^k - \frac{\pi_{2,j}^k}{\rho} - \frac{r_{2,j}^k}{d_2(i)}\Big\} \tag{6.32a}$$

$$s_{2,j}^{k+1} = \max\Big\{0, z_{2,j}^k - \frac{\pi_{3,j}^k}{\rho} - \frac{r_{3,j}^k}{d_3(i)}\Big\}. \tag{6.32b}$$

   (b) Exchange of z_j and $\pi_{1,i}$

   (c) For each constraint $i \in I_j = \{i; \exists A_{ip} \neq 0, p = \{j, j+N\}\}$, compute

$$\pi_{1,i}^{k+1} \hat{=} \pi_i^{k+1} = \frac{\rho}{d_1(i)} r_{1,i}^{k+1} + \pi_{1,i}^k \tag{6.33a}$$

$$\pi_{2,i}^{k+1} \hat{=} \pi_{i_{+N}}^{k+1} = \frac{\rho}{d_2(i)} r_{2,i}^{k+1} + \pi_{2,i}^k \tag{6.33b}$$

$$\pi_{3,i}^{k+1} \hat{=} \pi_{i_{+N+m}}^{k+1} = \frac{\rho}{d_3(i)} r_{3,i}^{k+1} + \pi_{3,i}^k \tag{6.33c}$$

   where, $d_1(i)$, $d_2(i)$ and $d_3(i)$ represent the degree of constraint i, $i{+}N$ and $i{+}N{+}m$ respectively; $r_{1,i}$,$r_{2,i}$ and $r_{3,i}$ are their residuals.

3. If the stopping criterion is not satisfied, increase k and go to 2). Otherwise, retain x_j^{k+1} and z_j^{k+1}.

Algorithm 1 is effectively distributed; for each agent j only local information $(f_j, L_{j,\cdot})$ and exchanges with the neighbors are required to compute its personal variables. We can also remark that, contrary to the consensus-based distributed ADMM [91], the relations between the agents are directly linked to the original constraints and not due to a reformulation of an originally centralized problem.

It is straightforward to see that the conditions for convergence of the ADMM algorithm are met provided that Assumption 1 is verified.

Theorem 2. *Let Assumption 1 hold. Then, the generated sequence*

$$\boldsymbol{u}^k = ([\boldsymbol{x}^k, \boldsymbol{z}^k, \boldsymbol{s}_1^k, \boldsymbol{s}_2^k])$$

for $k > 0$ converges to

$$\boldsymbol{u}^\star = [\boldsymbol{x}^\star; \boldsymbol{y}^\star; \boldsymbol{z}^\star, \boldsymbol{s}_1^\star, \boldsymbol{s}_2^\star]$$

where

$$(\boldsymbol{x}^\star, \boldsymbol{y}^\star, \boldsymbol{z}^\star)$$

is a solution of Problem (4.1).

This results from the relation linking $\boldsymbol{\zeta}$ to $\boldsymbol{u}$ and $\boldsymbol{v}$. Iterating on $\boldsymbol{\zeta}$ will lead to $\boldsymbol{\zeta}^*$ and intuitively we will converge to $\boldsymbol{u}^*$ and $\boldsymbol{v}^*$.

This algorithm can be implemented either as a fully distributed ADMM application which requires synchronization and communication between all the connected nodes (by aid of the results of Section 6.5); or as a centralized application which requires a central fusion central communicating with all the nodes and with high computational capacities. Next, we give the distributed ADMM application to the production sharing problem that was provided in the previous chapter.

The initial conditions $\boldsymbol{u}^0$ and $\boldsymbol{\pi}^0$ can be any convenient starting point such as a previous solution or flat start.

6.3 Synchronous DPS w/ADMM

In the case of a network of agents with multiple authorities or when the network is divided into multiple micro-networks, it is advised to use area-based distributed mechanisms to manage efficiently the network. Interestingly, the components of $\boldsymbol{\lambda}$ and $\boldsymbol{v}$ are linked either to a node or to an edge of network; thus, updating only a part of these components sums up to performing computations and exchanges only on a sub-graph.

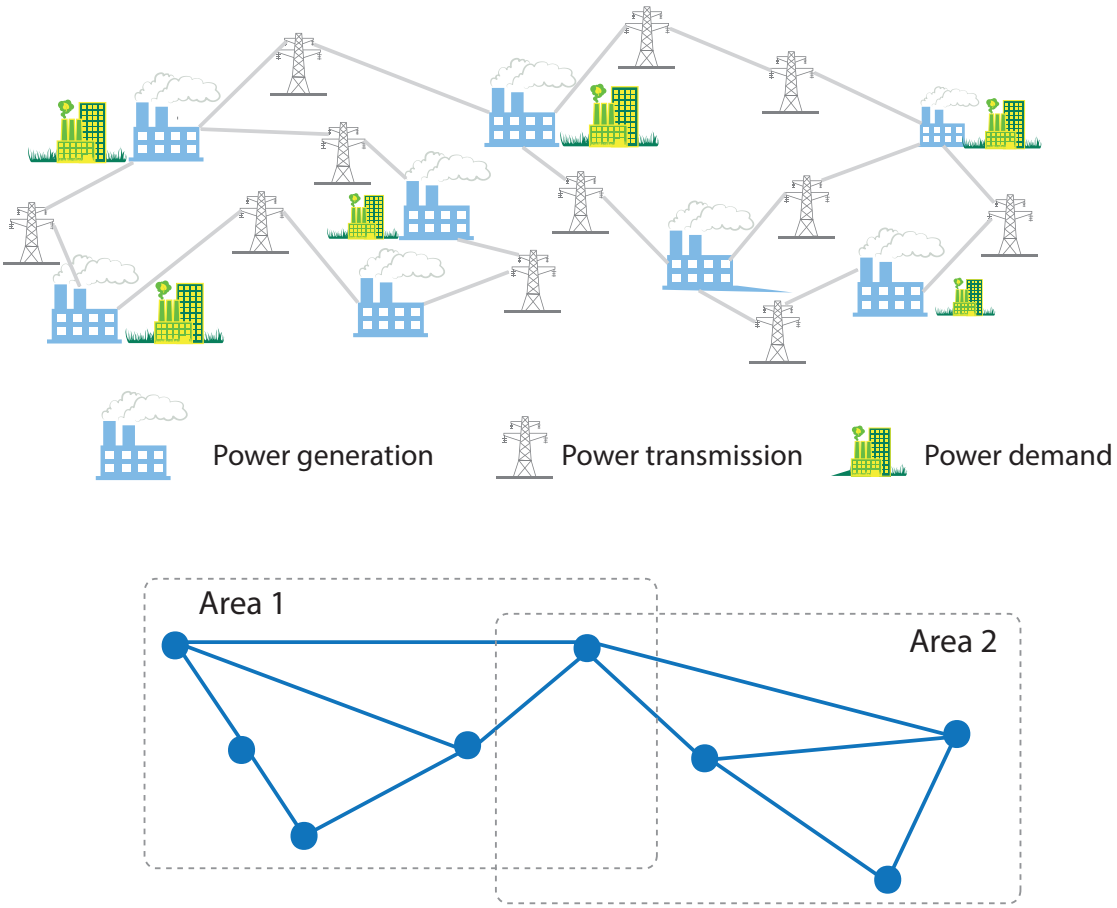

Figure 6.1: Power grid and its graph presentation divided into 2 overlapping areas.

Let us decompose the graph G into L connex areas A_ℓ, $l \in \{1, \ldots, L\}$ (see Fig. 6.1). Each area will act as a local processor exchanging data with its closest neighbors. Area A_ℓ is a graph with vertices $V_\ell \subset V$ and edges $E_\ell = \{\{i, j\}; (i, j) \in V_\ell^2\} \cap E$ such that the following assumptions hold true.

Assumption 3. *For any $l \in \{1, ..., L\}$, let $G(V_\ell) \triangleq (V_\ell, E_\ell)$ be the sub-graph of area A_ℓ, the following properties are assumed:*

1. $\bigcup_{l=1}^{L} V_\ell = V$,
2. $\bigcup_{l=1}^{L} G(V_\ell)$ *is connected.*

This means that in order to ensure convergence, the areas must overlap (*i.e.* have at least an agent in common, see Fig. 6.1), which can be restrictive.

At each iteration k, every area $A_l, l \in \{1, \ldots, L\}$ will apply equations (6.1a) and (6.1b) for the components of $\boldsymbol{u}$ and $\boldsymbol{v}$ that correspond to them. A communication step is then required before solving the components of (6.1c) assigned to the constraints of the nodes in V_l. This communication step is only performed between the neighboring areas sharing common nodes. It can be conducted through the shared nodes in an implicit manner, or through the processors of the overlapping areas. The synchronous distributed production sharing algorithm using ADMM is given as follows. This algorithm is obtained through similar calculations as in Algorithm 1. We only need to follow the same steps while considering the division of the network between multiple areas.

Algorithm 2: Synchronous DPS optimization w/ ADMM

1. Initialize $\boldsymbol{u}$ and $\boldsymbol{\pi}$ to the initial values $\boldsymbol{u}^0$ and $\boldsymbol{\pi}^0$.
2. At iteration k, each area A_l becomes operational, for each agent $j \in V_l$ the processor in this area performs the following steps:[2]
   (a) Updates x_j, z_j, $s_{1,j}$ and $s_{2,j}$ using equations (6.31a), (6.31b), (6.32a) and (6.32b).
   (b) Communicates the values z_j^{k+1} of border nodes to the neighboring areas.
   (c) Let $I_j = \{i; \exists A_{ip} \neq 0, p = \{j, j+N\}\}$ be the set of constraints corresponding to nodes $j \in V_l$. For each constraint $i \in I_j$, the processor computes $r_{1,i}^{k+1}$, $r_{2,i}^{k+1}, r_{3,i}^{k+1}$ and update $\pi_{1,i}^{k+1}$, $\pi_{2,i}^{k+1}$ and $\pi_{3,i}^{k+1}$ using equations (6.33a) to (6.33c) respectively.
3. If the stopping criterion is not satisfied, increase k and go to 2). Otherwise, retain x_j^{k+1} and z_j^{k+1}.

Algorithm 2 shares the same synchronization problem as Algorithm 1 that was given in the previous section. This property induces latency in the computation. All the processors are activated during every iteration and regular communication are required between all them. Next, we adopt an asynchronous version of this algorithm where only one portion of the network is updated in a given iteration.

[2] If however each node had a processor, then each node can perform the computation of its variables and the communication with its neighbors.

6.4 Randomized DPS w/ ADMM

6.4.1 Theoretical foundations

In the previous sections, we presented two algorithms for solving Problem (4.1) in a distributed manner over the graph associated with $\mathbf{L}$. However, it is not always possible nor appropriate to compute a full iteration of this algorithm. For example, some agents of the network can randomly fail to exchange or compute their updates. Also, it may be faster to solve the problem by taking into account only a random subset of the agents/links at each iteration in the spirit of *mini-batch* algorithms.

For these reasons, it is interesting to consider a randomized version of the previous algorithm where only some parts of the network are active at a given iteration.

This can be achieved by following the ADMM randomization scheme proposed in [77]. At each iteration k, this method consists of picking a random set of coordinates ξ^k (or equivalently a random subset of agents and links), and performing the updates of equations (6.1b) and (6.1c) only for the coordinates of ξ^k, the other ones being kept at their former value. As for the first update equation (6.1a), only the coordinates needed for the partial update of equations (6.1b) and (6.1c) are to be computed.

In this way, provided that the random coordinate selection sequence $(\xi^k)_{k>0}$ is i.i.d. and such that the selection probability is positive for every coordinate, the randomized algorithm converges almost surely to a sought solution [77] (see also [92] for refinements).

6.4.2 Randomized algorithm

As indicated previously, when we iterate $\boldsymbol{\zeta}^{k+1} = \mathcal{S}(\boldsymbol{\zeta}^k)$, we obtain an algorithm that imitates the behavior of the well-known synchronous ADMM algorithm.

In order to obtain an asynchronous algorithm, we endow this updating process with a random behavior. To this end, let $\boldsymbol{\zeta} = [\boldsymbol{\zeta}_1; \ldots; \boldsymbol{\zeta}_L]$ where $\boldsymbol{\zeta}_l = \boldsymbol{\lambda}_l + \rho \boldsymbol{v}_l$, $\boldsymbol{\lambda}_l$ and $\boldsymbol{v}_l$ are the vector containing the Lagrangian multipliers and the components of $\boldsymbol{v}$ that corresponds to the nodes and edges in area A_l. Suppose $\mathcal{S}(\boldsymbol{\zeta}) = [\mathcal{S}_1(\boldsymbol{\zeta}); \ldots; \mathcal{S}_L(\boldsymbol{\zeta})]$, we define for each area A_l, the operator $\hat{\mathcal{S}}_l : \mathcal{Y} \to \mathcal{Y}$ as:

$$\hat{\mathcal{S}}_l(\boldsymbol{\zeta}) = [\boldsymbol{\zeta}_1; \ldots; \boldsymbol{\zeta}_{l-1}; \mathcal{S}_l(\boldsymbol{\zeta}); \boldsymbol{\zeta}_{l+1}; \ldots; \boldsymbol{\zeta}_L)] \tag{6.34}$$

which implies to have only one area's process activated at every iteration.

After decomposing the network into *overlapping areas* such that Assumption 2 holds true, we assume the following assumption of the choice of ξ^k.

Assumption 4. *The area selection sequence $(\xi^k)_{k>0}$, valued in the set of the subsets of $\{1, ..., L\}$, is independent and identically distributed and such that $\forall \ell,\ \mathbb{P}[\ell \in \xi^1] > 0$.*

This assumption is needed to ensure that the area selection process is i.i.d. It is required to make sure that each component of $\boldsymbol{\lambda}$ or $\boldsymbol{v}$ has a positive update probability which is required conditions for the convergence [77].

The following theorem is applicable.

Theorem 3. *[77, Th. 2]: Take a firmly non-expansive operator $\mathcal{S} = [\mathcal{S}_1; \ldots; \mathcal{S}_L]$ with full domain on $\mathcal{Y}$ and a sequence of i.i.d. random variables $(\xi^k)_{k\in\mathbb{N}}$ such that Assumption 3 holds. Then, starting from any initial value $\boldsymbol{\zeta}^0$, the random iterates, $\boldsymbol{\zeta}^{k+1} = \hat{\mathcal{S}}_{\xi^{k+1}}(\boldsymbol{\zeta}^k)$ converges almost surely[3] to a random variable supported by $Fix(\mathcal{S})$ (when $Fix(\mathcal{S}) \neq \emptyset$).*

Thus, as can be concluded from Theorem 3, applying the area-based ADMM algorithm by random areas leads to a solution of the DPS problem (4.1).

The Asynchronous Distributed Production-Sharing algorithm with overlapping areas is stated in Algorithm 3. We will extend its application to the non-overlapping case in next section.

Mainly, at each iteration k the algorithm can be seen as applying equations (6.1a)-(6.1c) but keeping only the components related to the randomly chosen area of ξ^{k+1}. This is obtained through similar calculations to those used to obtain Algorithm 1. We only need to take into consideration the decomposition of the graph into multiple areas.

We remark that Algorithm 3 is effectively randomized in the sense that only the agents of the chosen areas perform computations and exchanges. The performance of this algorithm as well as details about the choice of the area selection sequence will be provided in Chapter 8. Furthermore, its convergence result is a direct consequence of Theorem 3.

[3] $\boldsymbol{\zeta}^k$ converges almost surely to $\boldsymbol{\zeta}$ if $\mathcal{P}_r(\lim_{k\to\infty} \boldsymbol{\zeta}^k = \boldsymbol{\zeta}) = 1$.

Algorithm 3: Asynchronous DPS optimization w/ ADMM

1. Initialize $\boldsymbol{u}$ and $\boldsymbol{\pi}$ to the initial values $\boldsymbol{u}^0$ and $\boldsymbol{\pi}^0$.
2. At iteration k, a random area A_{ξ^k} becomes operational, and for each agent $j \in V_{\xi^k}$ the following steps should be performed:
   (a) Update x_j, z_j, $s_{1,j}$ and $s_{2,j}$ using equations (6.31a), (6.31b), (6.32a) and (6.32b).
   (b) Communicate z_j^{k+1} to the neighboring nodes.
   (c) For each constraint $i \in I_j$, compute $r_{1,i}^{k+1}$, $r_{2,i}^{k+1}, r_{3,i}^{k+1}$ and update $\pi_{1,i}^{k+1}$, $\pi_{2,i}^{k+1}$ and $\pi_{3,i}^{k+1}$ using equations (6.33a) to (6.33c).
3. If the stopping criterion is not satisfied, increase k and go to 2). Otherwise, retain x_j^{k+1} and z_j^{k+1}.

Theorem 4. *Let Assumptions 1 and 3 hold. Then, the sequence* $(\boldsymbol{u}^k)_{k>0} = ([\boldsymbol{x}^k, \boldsymbol{y}^k, \boldsymbol{z}^k])_{k>0}$ *generated by* Asynchronous DPS optimization w/ ADMM *converges almost surely to* $\boldsymbol{u}^\star = [\boldsymbol{x}^\star; \boldsymbol{y}^\star; \boldsymbol{z}^\star]$ *where* $(\boldsymbol{x}^\star, \boldsymbol{y}^\star, \boldsymbol{z}^\star)$ *is a solution of Problem* (4.1).

The application of the asynchronous DPS algorithm using ADMM can be implemented either as a case of random failures within the network or in a way to reduce the computation and communication by turning areas on and off. Both schemes are implemented in Chapter 8. In the former, from an iteration to the following, nodes fail to update their variables which lead to a new graph. In the latter, the network is divided from the beginning into multiple areas that are randomly chosen to perform the update of their variables. In practice, this will lead into an efficient use of energy and less communication between the processors. Additionally, processor has more idle time and the number of computations they have to perform is reduced and more energy can be saved for other applications/tasks.

As mentioned before, the convergence result is limited to the case where these areas overlap. This is equivalent to having some agents falling under the authority of multiple areas. In power grid networks, this may result in conflicts in decision making as a coordination is required between these authorities. In the following section, we extend our algorithms to the case of non-overlapping areas by introducing dummy nodes between the areas. Subsequently, the areas become more independent as every agent refers to only one area.

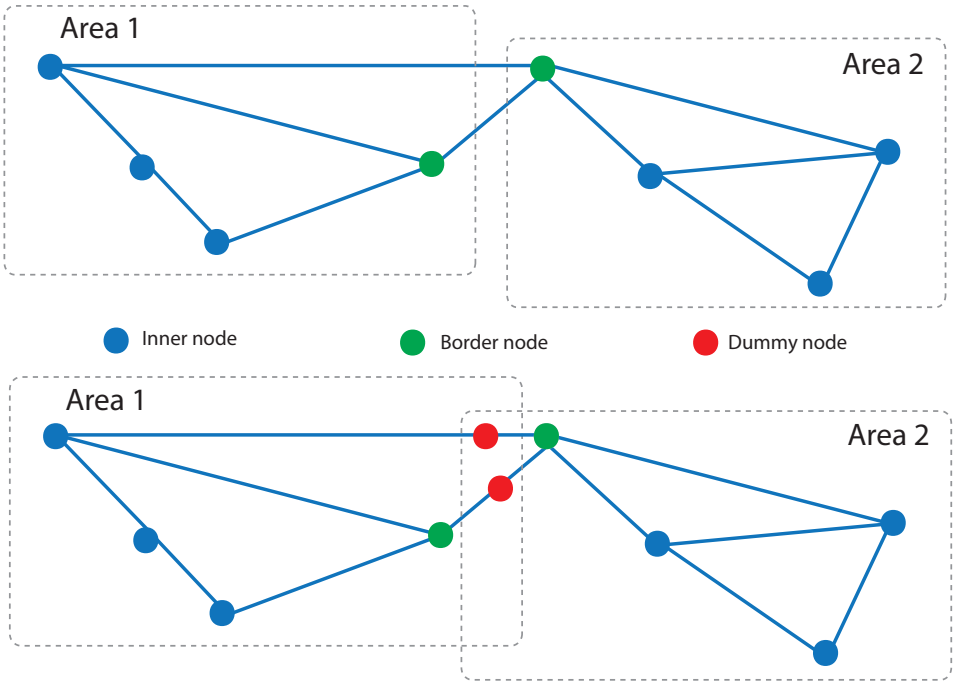

Figure 6.2: Network of 2 non-overlapping areas converted to overlapping network.

6.5 Extension to the non-overlapping areas case

Because imposing overlapping areas in a power network structure may lead to practical management problems, and inspired by the method of passing adjacent variables [93] that allows to convert networks of non-overlapping and independent areas into a network of overlapping areas, we extend the asynchronous distributed ADMM to cover this context of non-overlapping areas.

In this section, we extend our previous results to the case where non-overlapping areas are activated at each iteration. We focus here on the aforementioned DPS algorithm but the reasoning below can be easily adapted to a large class of distributed ADMM-based algorithms.

As opposed to Assumption 3, we divide here the graph G (linked to Laplacian $\mathbf{L}$ as described in the previous section) into L strictly separated areas A_ℓ, $\ell \in \{1, \ldots, L\}$. Every node belongs to exactly one area. A node will be called a *border node* if and only if it is connected to a node in a different area, and an *inner node* otherwise.

Starting from the initial Production-Sharing problem (4.1), we aim at decomposing it into L sub-problems, each associated to an area A_ℓ. Obviously, some constraints couple multiple areas together and thus cannot be assigned to one area. These coupling constraints are related to the connections between border nodes. Inspired by the method of passing adjacent variables [93], we propose an approach that transforms our problem so that only the agents of a given area are active at each iteration. For this, we add a *dummy node* between each pair of connected border nodes of different areas as shown in Fig. 6.2. These dummy nodes are not associated with any cost function but only serve to rewrite the constraints coupling adjacent areas using a *shared variable*, that will have to be exchanged as we will see later.

Thus, the changes to be applied to the initial optimization problem are:

- Add chosen dummy nodes;
- Rewrite the coupling constraints accordingly.

Practically, the problem becomes

Non-overlapping DPS Problem

$$\begin{aligned} \min_{\mathring{\boldsymbol{x}},\mathring{\boldsymbol{y}},\mathring{\boldsymbol{z}}\in\mathbb{R}^{\mathring{N}}} \quad & \textstyle\sum_{j\in\mathring{V}} \mathring{f}_j(\mathring{x}_j) \\ \text{subject to} \quad & \forall j \in \mathring{V}, \mathring{\underline{x}}_j \leq \mathring{x}_j \leq \mathring{\overline{x}}_j \\ & \begin{cases} \mathring{\boldsymbol{y}} = \mathring{\boldsymbol{x}} - \mathring{\boldsymbol{d}} \\ \mathring{\boldsymbol{y}} = \mathring{\mathbf{L}}\mathring{\boldsymbol{z}} \\ \mathring{\underline{\boldsymbol{p}}} \leq \mathring{\mathbf{M}}\mathring{\boldsymbol{z}} \leq \mathring{\overline{\boldsymbol{p}}} \end{cases} \end{aligned} \tag{6.35}$$

where,

- $\mathring{V} = V \bigcup \bullet{V}$ is the new set of $\mathring{N}$ agents composed of the set of the N original agents V plus the set of the $\bullet{N}$ dummy agents $\bullet{V}$;
- for all $j \in V$, $\mathring{f}_j$, $\mathring{\underline{x}}_j$, $\mathring{\overline{x}}_j$, and $\mathring{\boldsymbol{d}}_j$ are respectively equal to their original values f_j, $\underline{x_j}$, $\overline{x_j}$, and $\boldsymbol{d}_j$;
- for all $d \in \bullet{V}$, $\mathring{f}_d \equiv 0$, $\mathring{\underline{x}}_d = \mathring{\overline{x}}_d = 0$, and $\mathring{\boldsymbol{d}}_d = 0$;
- for all original links $i \sim j \in E$, the values of $\mathring{\underline{p}}$ and $\mathring{\overline{p}}$ are not modified;

– for all new links $i \sim d$, the values of $\overset{\circ}{\underline{p}}$ and $\overset{\circ}{\overline{p}}$ are equal to those of the corresponding tie-line in the non-overlapping network.

The design of $\overset{\circ}{\mathbf{L}}$ from $\mathbf{L}$ and $\overset{\circ}{\mathbf{M}}$ from $\mathbf{M}$ is given in the following Lemma and explained next.

Lemma 7. *The mapping between the non-overlapping problem complicating constraints and the modified overlapping problem is given by:*

- *If $i \sim j$ is a tie-line,*

$$\overset{\circ}{L}_{jj} = L_{jj} + \frac{1}{2}\sum_{d\sim j} \overset{\circ}{L}_{jd}, \quad \overset{\circ}{L}_{jd} = -2L_{ij}, \quad \overset{\circ}{L}_{dd} = 4L_{ij},$$

$$\overset{\circ}{M}_{jd} = -2M_{ij}.$$

- *If i and j belong to the same area,*

$$\overset{\circ}{L}_{ij} = L_{ij},$$

$$\overset{\circ}{M}_{ij} = M_{ij}.$$

where d is the dummy node that was inserted on the tie-line linking nodes i and j. These two nodes belonged to different areas in the non-overlapping areas division. L_{ij} and L_{jj} are the entries related to the original network Laplacian $\mathbf{L}$. $\overset{\circ}{L}_{ij}$ and $\overset{\circ}{L}_{jj}$ are the entries related to the modified network Laplacian $\overset{\circ}{\mathbf{L}}$.

Proof: In the original problem, the constraint related to $\mathbf{L}$ writes for all $j \in V$, $y_j = \sum_{i\in V} L_{ji}z_i$. Furthermore, if j belongs to area ℓ, this can be rewritten $y_j = \sum_{i\in A_\ell} L_{ji}z_i + \sum_{k\in V\setminus A_\ell} L_{jk}z_k$ where the second term represents the coupling relations we want to eliminate thanks to the addition of dummy nodes.

In the new problem, the constraint related to $\overset{\circ}{\mathbf{L}}$ is written for all nodes $j \in A_\ell$ as $\overset{\circ}{y}_j = \sum_{i\in A_\ell\setminus j} \overset{\circ}{L}_{ji}\overset{\circ}{z}_i + \sum_{d\sim j} \overset{\circ}{L}_{jd}\overset{\circ}{z}_d + \overset{\circ}{L}_{jj}\overset{\circ}{z}_j$. For all $i, j \in V$ belonging to the same area, we have $\overset{\circ}{L}_{ji} = L_{ji}$.

Noticing that $\overset{\circ}{y}_d = 0$ for all $d \in \overset{\bullet}{V}$, we get that $0 = \overset{\circ}{L}_{dd}\overset{\circ}{z}_d + \overset{\circ}{L}_{di}\overset{\circ}{z}_i + \overset{\circ}{L}_{dj}\overset{\circ}{z}_j = \overset{\circ}{L}_{dd}\overset{\circ}{z}_d + \overset{\circ}{L}_{id}\overset{\circ}{z}_i + \overset{\circ}{L}_{jd}\overset{\circ}{z}_j$, where i and j are the two original nodes between which the dummy node was inserted (and the second equality is due to the fact that

$\mathbf{L}$ needs to remain symmetric). Node d serves as a dummy node which does not generate or consume resources. The following equality should hold for each dummy node inserted, $\overset{\circ}{L}_{dj}(\overset{\circ}{z}_j - \overset{\circ}{z}_d) + \overset{\circ}{L}_{di}(\overset{\circ}{z}_i - \overset{\circ}{z}_d) = 0$. We assume the symmetry on the edges linking d to i and j, thus $\overset{\circ}{L}_{id} = \overset{\circ}{L}_{jd} = \overset{\circ}{L}_{di} = \overset{\circ}{L}_{dj}$. We obtain $\overset{\circ}{L}_{dd} = -2\overset{\circ}{L}_{dj}$. We can rewrite $\overset{\circ}{y}_j$ as $\overset{\circ}{y}_j = \sum_{k\in A_\ell \setminus j} \overset{\circ}{L}_{jk}\overset{\circ}{z}_k + (\overset{\circ}{L}_{jj} - \frac{1}{2}\sum_{d\sim j} \overset{\circ}{L}_{jd})\overset{\circ}{z}_j - \frac{1}{2}\sum_{i\sim j, d\sim j} \overset{\circ}{L}_{dj}\overset{\circ}{z}_i$.

Comparing this expression of $\overset{\circ}{y}_j$ with the previous one, we obtain,

$$\overset{\circ}{L}_{jj} = L_{jj} + \frac{1}{2}\sum_{d\sim j} \overset{\circ}{L}_{jd}, \quad \overset{\circ}{L}_{jd} = -2L_{ij}, \quad \overset{\circ}{L}_{dd} = 4L_{ij}.$$

Matrix $\mathbf{M}$ is obtained from $\mathbf{L}$. The same follows for the modified matrix $\overset{\circ}{\mathbf{M}}$. Thus, we have:

$$\overset{\circ}{M}_{jd} = -2M_{ij}, \text{ if } i \sim j \text{ is a tie-line,}$$

$$\overset{\circ}{M}_{ij} = M_{ij}, \text{ if } i \text{ and } j \text{ belong to the same area.}$$

With this formulation, Problems (4.1) and (6.35) are equivalent. Following the reasoning of the previous sections on the modified problem, one can derive a new algorithm corresponding to the previous one but with the modified functions and matrices. A solution of the original problem is then extracted from a solution of the modified problem by omitting the entries related to the dummy nodes. One can remark that in the new algorithm, when an area is updated, it only needs local information plus the value of the variables corresponding to the dummy nodes. An iteration is thus now composed of two parts: i) local computations and exchanges in the selected area; ii) communication of border-related values to the adjacent areas.

6.6 Conclusion

We provided in this chapter the detailed results of applying ADMM to the DPS problem. First, we considered the case where the network is divided into multiple overlapping areas and we presented the distributed ADMM application. We solved the synchronous problem of this method by presenting an asynchronous application of the ADMM and we proved its convergence. All of these applications required having some nodes shared between the areas.

We concluded this chapter by extending the application of the synchronous and asynchronous ADMM to the case of a network with non-overlapping areas. The areas are now independent, all he nodes belong to only one area. Their values are updated when the corresponding area is activated.

Chapter 7

Integration of Renewable sources and Storage devices

7.1 Introduction

The objective of this chapter is to formulate a distributed control algorithm for cooperation of storage operation in a power network under the framework of distributed generation (DG) units. As the number of DG units in a power network scales up, the implementation of centralized control policies start to become non feasible owing to the tremendous amount of information that must be exchanged in the power network. Therefore, under a realistic setting, the DGs will have to operate autonomously with limited information exchange between them. Moreover, they cannot see the impact of their actions on the entire power grid. Therefore, decentralized control strategies must be designed that ensure the stability of the grid.

This is a joint work with Dr. Subhash Lakshminarayana where we intend to formulate decentralized control policies for storage operation in a power network by combining techniques from Lyapunov optimization [94, 95] and the ADMM framework.

7.2 System Model

We consider the same power network consisting of a set V of N buses (nodes) and m branches connecting the buses denoted by the set E. We add the following modifications to the problem described in Chapter 4.

- The variables are now per time slot:

  - The conventional power generation at node j during time t is denoted $p_j^G[t]$,it is bounded by $\underline{p_j^G}[t]$ and $\overline{p_j^G}[t]$.
  - The voltage phase angle at bus j is given by $\theta_j[t]$.
  - The demand at node j during time slot t, $p_j^D[t]$, is assumed to be inelastic.
  - The power flow on branch (j, i) is denoted $p_{ji}[t]$.
  - The cost of conventional generation denoted by the function $f_j(p_j^G[t])$ is assumed to be a quadratic function given as [1]

$$f_j(p_j^G[t]) = c_j p_j^G[t] + c'_j p_j^{G^2}[t], \tag{7.1}$$

    where c_j and c'_j are constants that depend on the generation technology at bus j.

- Every bus j can harvest $X_j[t]$ units of energy from renewable generation during time t.
  - $X_j[t]$ is assumed to evolve according to a Markovian process.
  - The cost of generation of the renewable energy is assumed to be zero.

- Each bus can be equipped with a battery of storage capacity $E_{j,\max}$ units.
  - The battery charging and discharging operations at bus j are represented by $Y_{i,\text{ch}}[t]$ and $Y_{i,\text{dis}}[t]$ respectively.
  - $Y_{i,\text{ch}}[t]$ and $Y_{i,\text{dis}}[t]$ are bounded by $Y_{\text{ch}}^{\max}$ and $Y_{\text{dis}}^{\max}$.

$$0 \leq Y_{j,\text{ch}}[t] \leq Y_{\text{ch}}^{\max}, \qquad \forall t, \quad j \in V \tag{7.2}$$

$$0 \leq Y_{j,\text{dis}}[t] \leq Y_{\text{dis}}^{\max}, \qquad \forall t, \quad j \in V. \tag{7.3}$$

  - The charging and discharging efficiencies are modeled using $\eta_{\text{ch}} \leq 1$ and $\eta_{\text{dis}} \geq 1$.
  - The energy level in the battery E_j at node j is constrained as follows:

$$0 \leq E_i[t] \leq Emax \tag{7.4}$$

  - The energy level in the battery evolves as

$$E_j[t+1] = E_j[t] + \eta_{\text{ch}} Y_{j,\text{ch}}[t] - \eta_{\text{dis}} Y_{j,\text{dis}}[t] \quad j \in V. \tag{7.5}$$

  - The battery discharge decision is constrained by the energy available at bus j:

$$\eta_{\text{dis}} Y_{j,\text{dis}}[t] \leq E_j[t], \qquad \forall t, \quad j \in V. \tag{7.6}$$

7.3 Problem Formulation

The energy availability (7.3) and the battery discharge constraint (7.6) can be combined as follows:

$$Y_{j,\text{dis}}[t] \leq \min\left(\frac{E_j[t]}{\eta_{\text{dis}}}, Y_{\text{dis}}^{\max}\right) \quad j \in V. \tag{7.7}$$

Similarly, combining (7.2) and (7.4) leads to:

$$Y_{j,\text{ch}}[t] \leq \min\left(\frac{E_{\max} - E_j[t]}{\eta_{\text{ch}}}, Y_{\text{ch}}^{\max}\right) \quad j \in V. \tag{7.8}$$

We make the following practical assumption on the battery capacity:

$$\eta_{\text{dis}} Y_{\text{dis}}^{\max} < E_{\max}.$$

We also consider that each battery has the same charging and discharging efficiency

$$\eta_{\text{dis}} = \eta_{\text{ch}}$$

and we use the following variable to denote the battery decision at time slot t,

$$Y_j[t] = Y_{j,\text{ch}}[t] - Y_{j,\text{dis}}[t].$$

The objective of the controller is to design the system parameters in order to minimize the time average cost of energy generation subject to the DC power flow equations and battery operational constraints, stated as:

$$\min_{\substack{p_j^G, Y_j, H_j, \theta_j \\ j \in V}} \quad \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\left[\sum_{j=1}^{N} f_j(p_j^G[t])\right]$$

$$\text{subject to} \quad \begin{cases} E_j[t+1] = E_j[t] + Y_j[t] \\ p_j^D[t] - H_j[t] - p_j^G[t] + Y_j[t] + \sum_{i\sim j} p_{ji}[t] = 0, \\ H_j[t] \leq X_j[t], \\ \underline{p_{ij}} \leq p_{ij}[t] \leq \overline{p_{ij}}, \\ -\min\left(\frac{E_j[t]}{\eta_{\text{dis}}}, Y_{\text{dis}}^{\max}\right) \leq Y_j[t] \leq \min\left(\frac{E_{\max}-E_j[t]}{\eta_{\text{ch}}}, Y_{\text{ch}}^{\max}\right). \end{cases} \tag{7.9}$$

$p_{ji}[t] = B_{ji}(\theta_j[t] - \theta_i[t])$ represents the power transmitted from node j to node i or the inverse.

We denote the minimum time average cost of (7.9) over all feasible control policies by $f_{\min}$. The optimization problem (7.9) is in the form of a stochastic

dynamic programming problem. Solving this problem using conventional dynamic programming based techniques can be computationally complex, especially when the state space of the system is large. In what follows, we use the technique of Lyapunov optimization to develop a low complexity online solution to this problem [94, 95].

7.4 Algorithm Design by Lyapunov Optimization

In order to solve (7.9) using the technique of Lyapunov optimization, we first introduce an approximate version of this problem in which we relax all the constraints associated with the battery, stated as follows [96]:

$$\min_{\substack{p_j^G, Y_j, H_j, \theta_j \\ j \in V}} \quad \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\left[\sum_{j=1}^{N} f_j(p_j^G[t])\right]$$

$$\text{subject to} \quad \begin{cases} \overline{Y}_{j,\text{ch}} \leq \overline{Y}_{j,\text{dis}} \\ E_j[t+1] = E_j[t] + Y_j[t] \\ p_j^D[t] - H_j[t] - p_j^G[t] + Y_{j,}[t] + \sum_{i\sim j} p_{ji}[t] = 0, \\ H_j[t] \leq X_j[t], \\ \underline{p_{ij}} \leq p_{ij}[t] \leq \overline{p_{ij}}, \\ -\min\left(\frac{E_j[t]}{\eta_{\text{dis}}}, Y_{\text{dis}}^{\max}\right) \leq Y_j[t] \leq \min\left(\frac{E_{\max} - E_j[t]}{\eta_{\text{ch}}}, Y_{\text{ch}}^{\max}\right) \end{cases} \tag{7.10}$$

where,

$$\overline{Y}_{j,\text{ch}} = \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}[Y_{j,\text{ch}}[t]]$$

and $\overline{Y}_{j,\text{dis}}$ is defined similarly.

We subsequently address (7.10) as the *relaxed problem*. Note that in the relaxed problem, the constraint $\overline{Y}_{j,\text{ch}} \leq \overline{Y}_{j,\text{dis}}$ corresponds to the stability of the virtual energy queue in (7.5) associated with the battery, and represents the case with infinite energy storage capacity.

In what follows, we solve the relaxed problem using the Lyapunov optimization technique. However, while designing the control algorithm, we introduce a perturbation parameter α as well as a control parameter Q. By carefully tuning these two parameters, we ensure that the algorithm developed for the relaxed problem is feasible for the original problem as well. We

subsequently omit the details of the algorithm design. In the proposed algorithm, the storage operations during each time slot can be computed as a solution to the following optimization problem.

It is worth noting that solving (7.11) requires the knowledge of only the current state of the system. Moreover, it does not suffer from the *curse of scalability* as in the case of dynamic programming based solutions.

Algorithm 4 [95]: Lyapunov Optimization for DC-OPF with storage devices and renewable energy sources

---

1. Initialize $t = 0$ and $E_j[0] = 0 \ \forall j$.

2. For each $j \in V$, compute $Y_j[t], p_j^G[t], \theta_j[t]$ and $H_j[t]$ as the solution to the following linear programming problem:[1]

$$\min_{\substack{p_j^G, Y_j, H_j, \theta_j \\ j \in V}} \quad \sum_{j=1}^{N} (E_j[t] - E_{\max}) Y_j[t] + Q \sum_{j=1}^{N} f_i(p_j^G[t])$$

$$\text{subject to} \quad \begin{cases} -\min\left(\frac{E_j[t]}{\eta_{\text{dis}}}, Y_{\text{dis}}^{\max}\right) \leq Y_j[t] \leq \min\left(\frac{E_{\max} - E_j[t]}{\eta_{\text{ch}}}, Y_{\text{ch}}^{\max}\right) \\ p_j^D[t] - H_j[t] - p_j^G[t] + Y_j[t] + \sum_{i \sim j} p_{ji}[t] = 0, \\ H_j[t] \leq X_j[t], \\ \underline{p_{ij}} \leq p_{ij}[t] \leq \overline{p_{ij}}, \end{cases} \tag{7.11}$$

3. Update $t = t + 1$. If $t = T_{\max}$ then terminate. Otherwise, update the battery level using

$$E_j[t+1] = E_j[t] + Y_j[t], \ \forall \, j \in V.$$

and return to step 2.

We now state the result on the algorithm performance analysis.

[1] In this thesis, $\underline{p_{ij}}$ is chosen equal to $-\overline{p_{ij}}$

Theorem 5. *[95] By choosing the value of the parameters α and Q as*

$$\alpha = (\eta_{ch} Y_{ch}^{\max})^2; Q = \frac{\eta_{ch} E_{\max} - \eta_{dis} Y_{dis}^{\max}}{c_{\max} + c'_{\max} \min(d_{\max}, Y_{dis}^{\max})}, \tag{7.12}$$

Algorithm 4 can be made feasible for (7.9). Furthermore, the time average cost function achieved by this algorithm satisfies:

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E}\left[\sum\nolimits_{ji=1}^{N} f_j(p_j^G[t])\right] \leq f_{\min} + \frac{\tilde{B}}{Q}, \tag{7.13}$$

where, $\tilde{B} < \infty$ is a constant, $c_{\max} = \max_{j\in V} c_j$ and $c'_{\max} = \max_{j\in V} c'_j$.

Theorem 5 implies that the performance of the algorithm is at a bounded distance from the optimal value, where the bound is determined by the value of parameter Q. However as evident from (7.12), the value of the parameter Q depends on the battery capacity $E_{\max}$. Therefore, for large $E_{\max}$, performance of the algorithm can be made arbitrarily close to $f_{\min}$.

7.5 Distributed formulation of the problem

Let x_j denotes the power generated at node j and z_j its voltage phase angle. We drop the time parameter t and we start by rewriting problem (7.11) using global vectorial presentation. We regroup the variables as follows:

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix}, \boldsymbol{z} = \begin{bmatrix} z_1 \\ \vdots \\ z_N \end{bmatrix}, \boldsymbol{Y} = \begin{bmatrix} Y_1 \\ \vdots \\ Y_N \end{bmatrix}, \text{and } \boldsymbol{H} = \begin{bmatrix} , H_1 \\ \vdots \\ H_N \end{bmatrix}.$$

We also convert the inequalities related to power line flow into equality constraints using two set of slack variables:

$$\boldsymbol{s_1} = \begin{bmatrix} s_{1,1} \\ \vdots \\ s_{1,m} \end{bmatrix} \text{ and } \boldsymbol{s_2} = \begin{bmatrix} s_{2,1} \\ \vdots \\ s_{2,m} \end{bmatrix},$$

where m is the number of branches.

Let $\boldsymbol{u}$ be the vector incorporating the network's global variables:

$$\boldsymbol{u} = \begin{bmatrix} \boldsymbol{G} \\ \boldsymbol{\theta} \\ \boldsymbol{Y} \\ \boldsymbol{H} \\ \boldsymbol{S_1} \\ \boldsymbol{S_2} \end{bmatrix}. \tag{7.14}$$

We regroup the power flow equations using the following matrix:

$$\boldsymbol{A} = \begin{bmatrix} \mathbf{I} & \mathbf{L} & -\mathbf{I} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & -\mathbf{M} & \mathbf{0} & \mathbf{0} & \mathbf{o} & \mathbf{I} \end{bmatrix}, \tag{7.15}$$

and the vector:

$$\tilde{\boldsymbol{d}} = \begin{bmatrix} p_1^D \\ \vdots \\ p_N^D \\ \overline{p_{ij}} \\ \vdots \\ \overline{p_{ij}} \end{bmatrix}. \tag{7.16}$$

$\boldsymbol{A}$ is an $(N + 2m) \times (4N + 2m)$ matrix, and $\tilde{\boldsymbol{d}} \in \mathbb{R}^{(N+2m)}$. As explained in Chapter 4, equations $B_{j,i}(\theta_j - \theta_i)$ for each branch (i, j) are used to create the weighted branch-bus incidence matrix $\boldsymbol{M}$.

Following the reasoning in Chapter 4, let F be the new cost function corresponding to the function to minimize in (7.11) applied to $\boldsymbol{u}$ and encompassing the box constraints on the generated power. Define the matrix $\tilde{\mathbf{A}}$ of size $N' \times (4N + 2m)$, where $N' = (N + 2m)(4N + 2m)$

$$\tilde{\mathbf{A}} = \begin{bmatrix} \operatorname{diag}([\mathbf{A}]_1) \\ \vdots \\ \operatorname{diag}([\mathbf{A}]_{N+2m}) \end{bmatrix} \tag{7.17}$$

and the indicator function

$$\begin{array}{rccl} G: & \mathbb{R}^{N'} & \longrightarrow & (-\infty, +\infty] \\ & \boldsymbol{v} & \longmapsto & \begin{cases} 0 \quad \text{if } \forall i = 1 \displaystyle\sum_{j=4(i-1)N+2(i-1)m+1}^{4Ni+2mi} v_j = \tilde{d}_j \\ +\infty \quad \text{otherwise} \end{cases} \end{array} \tag{7.18}$$

The problem can be reformulated as the *DPS Problem*

$$\min_{\boldsymbol{u}\in\mathbb{R}^{3N}} \quad F(\boldsymbol{u}) + G(\tilde{\mathbf{A}}\boldsymbol{u}) \tag{7.19}$$

To this reformulated problem we can apply the distributed ADMM algorithm. As previously explained in Chapters 4 and 5, an iteration of the ADMM algorithm leads to the following equations:

$$\boldsymbol{u}^{k+1} = \underset{\boldsymbol{u}}{\operatorname{argmin}} \left\{ F(\boldsymbol{u}) + \frac{\rho}{2} \left\| \tilde{\mathbf{A}}\boldsymbol{u} - \boldsymbol{v}^k + \frac{\boldsymbol{\lambda}^k}{\rho} \right\| \right\}^2 \tag{7.20}$$

$$\boldsymbol{v}^{k+1} = \underset{\boldsymbol{v}}{\operatorname{argmin}} \left\{ G(\boldsymbol{v}) + \frac{\rho}{2} \left\| \tilde{\mathbf{A}}\boldsymbol{u}^{k+1} - \boldsymbol{v} + \frac{\boldsymbol{\lambda}^k}{\rho} \right\| \right\}^2 \tag{7.21}$$

$$\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + \rho \left(\tilde{\mathbf{A}}\boldsymbol{u}^{k+1} - \boldsymbol{v}^{k+1} \right) \tag{7.22}$$

where $\rho > 0$ is a free hyper-parameter.

Let r_i^{k+1} be the residual of the i^{th} constraint,

$$\begin{aligned} r_i^{k+1} = &\sum_{j=1}^{N} A_{ij} x_j^{k+1} + A_{ij_{+N}} z_j^{k+1} + A_{ij_{+2N}} Y_j^{k+1} + A_{ij_{+3N}} H_j^{k+1} \\ &+ \sum_{j=1}^{m} A_{ij_{+4N}} s_{1,j}^{k+1} + A_{ij_{+4N+m}} s_{2,j}^{k+1} - \tilde{d}_j, \end{aligned} \tag{7.23}$$

π_i be the Lagrangian multipliers assigned to this constraint

$$\pi_i^{k+1} = \pi_i^k + \frac{\rho}{d(i)} r_i^{k+1} \tag{7.24}$$

and d_i be its degree.

After several simplifications, we obtain the following set of equations for problem (7.11). The distributed Optimization result obtained by combining Lyapunov optimization and fully distributed ADMM is given in Algorithm 6.

For sake of simplicity, we divided these vectors as follows:

$$\begin{aligned}
\pi_{1,i} &= \pi_i, \forall i = 1, \ldots, N \\
\pi_{2,i} &= \pi_{i+N}, \forall i = 1, \ldots, m \\
\pi_{3,i} &= \pi_{i+N+m}, \forall i = 1, \ldots, m \\
r_{1,i} &= r_i, \forall i = 1, \ldots, N \\
r_{2,i} &= r_{i+N}, \forall i = 1, \ldots, m \\
r_{3,i} &= r_{i+N+m}, \forall i = 1, \ldots, m \\
d_1(i) &= \tilde{d}(i), \forall i = 1, \ldots, N \\
d_2(i) &= \tilde{d}(i+N), \forall i = 1, \ldots, m \\
d_3(i) &= \tilde{d}(i+N+m), \forall i = 1, \ldots, m.
\end{aligned}$$

The Lagrangian multipliers $\pi_{1,i}^{k+1}, \pi_{2,i}^{k+1}$ and $\pi_{3,i}^{k+1}$ are updated using the following equations:

$$\pi_{1,i}^{k+1} \hat{=} \pi_i^{k+1} = \frac{\rho}{d_1(i)} r_{1,i}^{k+1} + \pi_{1,i}^k \tag{7.25a}$$

$$\pi_{2,i}^{k+1} \hat{=} \pi_{i_{+N}}^{k+1} = \frac{\rho}{d_2(i)} r_{2,i}^{k+1} + \pi_{2,i}^k \tag{7.25b}$$

$$\pi_{3,i}^{k+1} \hat{=} \pi_{i_{+N+m}}^{k+1} = \frac{\rho}{d_3(i)} r_{3,i}^{k+1} + \pi_{3,i}^k \tag{7.25c}$$

where, $d_1(i)$, $d_2(i)$ and $d_3(i)$ represent the degree of constraint i, $i+N$ and $i+N+m$ respectively; $r_{1,i}$,$r_{2,i}$ and $r_{3,i}$ are their residuals.

Algorithm 6: Lyapunov optimization and fully distributed ADMM Production-Sharing optimization w/ renewable and storage devices

1. Initialize $\boldsymbol{u}$ and $\boldsymbol{\pi}$ to the initial values $\boldsymbol{u}^0$ and $\boldsymbol{\pi}^0$.

2. At iteration k:

   (a) For every agent $j = 1, \ldots, N$ update its variables using:

$$x_j^{k+1} = \frac{\rho}{\rho + 2Qc_j'} x_j^k - \frac{1}{\rho + 2Qc_j'} \Big(Qc_j + \pi_{1,j}^k + \frac{\rho}{d_1(j)} r_{1,j}^k \Big) \tag{7.26a}$$

$$z_j^{k+1} = z_j^k - \frac{1}{\sum_{i=1}^N L_{ij}^2 + 2\sum_{i=1}^m M_{ij}^2} \Bigg(\sum_{i=1}^N L_{ij} \Big(\frac{\pi_{1,i}^k}{\rho} + \frac{r_{1,i}^k}{d_1(i)} \Big) + \sum_{i=1}^m M_{ij} \Big(\frac{\pi_{2,i}^k}{\rho} + \frac{r_{2,i}^k}{d_2(i)} \Big) - \sum_{i=1}^m M_{ij} \Big(\frac{\pi_{3,i}^k}{\rho} + \frac{r_{3,i}^k}{d_3(i)} \Big) \Bigg) \tag{7.26b}$$

$$Y_j^{k+1} = Y_j^k + \frac{\pi_{1,j}^k - \alpha_j \eta_j}{\rho} + \frac{r_{1,j}^k}{d_1(j)} \tag{7.26c}$$

$$H_j^{k+1} = H_j^k - \frac{\pi_{1,j}^k}{\rho} - \frac{r_{1,j}^k}{d_1(j)} \tag{7.26d}$$

   For each connection $j = 1, \ldots, m$, the corresponding nodes update $s_{1,j}$ and $s_{2,j}$ using:

$$s_{1,j}^{k+1} = \max \Big\{ 0, s_{1,j}^k - \frac{\pi_{2,j}^k}{\rho} - \frac{r_{2,j}^k}{d_2(j)} \Big\} \tag{7.27a}$$

$$s_{2,j}^{k+1} = \max \Big\{ 0, s_{2,j}^k - \frac{\pi_{3,j}^k}{\rho} - \frac{r_{3,j}^k}{d_3(j)} \Big\} \tag{7.27b}$$

   (b) Exchange of z_j and $\pi_{1,i}$

   (c) For each constraint $i \in I_j = \{i; \exists A_{ip} \neq 0, p = \{j, j+N\}\}$, update the corresponding Lagrangian multipliers.

3. If the stopping criterion is not satisfied, increase k and go to 2). Otherwise, retain the computed values.

7.6 Conclusion

We have incorporated battery devices under a DC power flow framework. We devised an online algorithm for the problem based on the theory of Lyapunov optimization. Then, we solved this online problem in a decentralized fashion with only local computations and communication between neighboring nodes using task-based ADMM iterations.

Our solution can be useful for power grid designer to lay out optimal infrastructure in terms of storage and transmission lines to meet specific cost criteria.

Chapter 8

Implementations

8.1 Introduction

In this chapter, we implement the algorithms derived in this thesis on the DC-OPF problem, a special linear DC approximation of the OPF problem [79]. Simulations are first carried out on the conventional power grid. We considered the IEEE−30 bus test system [97] and the IEEE−118 bus test system for this problem. Then, we used the IEEE−6 bus test system for the implementation of ADMM on the DC-OPF with storage devices and renewable energy sources. We tried to show the objective and residual convergence properties of the algorithms. We also provided different scenarios and asynchronous applications in which the distributed ADMM algorithm was proven to be effective and scalable.

8.2 DC-OPF

We start by considering an electrical power grid of N nodes and m branches, and we focus on the DC-OPF problem. The set of nodes is denoted V, and E represents the set of edges/branches. The DC-OPF defined in 4.3 considered in this chapter can be written as the following:

$$\begin{aligned} &\min_{\boldsymbol{p}^G, \boldsymbol{p}^E, \boldsymbol{\theta} \in \mathbb{R}^N} \quad \sum_{j \in V} f_j(p_j^G) \\ &\text{subject to} \quad \begin{cases} \boldsymbol{p}^E = \boldsymbol{p}^G - \boldsymbol{p}^D \\ \boldsymbol{p}^E = \mathbf{L}^B \boldsymbol{\theta} \\ \underline{\boldsymbol{p}^G} \leq \boldsymbol{p}^G \leq \overline{\boldsymbol{p}^G} \\ |\mathbf{M}\boldsymbol{\theta}| \leq \overline{\boldsymbol{p}} \end{cases} \end{aligned} \tag{8.1}$$

$f_j(p_j^G)$ is a closed proper convex cost function on the power generated p_j^G at node $j \in V$. Each node has lower and upper limits on the generated power: $\underline{p^G} \leq p^G \leq \overline{p^G}$. These limits are reduced to zeroes if no generator is placed at node j. At each node j we need to meet a power demand p_j^D, either by generating power or by importing it from neighboring nodes using the transmission lines. Thermal limits are to be considered at each of these lines: $|\mathbf{M}\boldsymbol{\theta}| \leq \overline{\boldsymbol{p}}$ where $\boldsymbol{\theta}$ is the vector of all the voltage angles and $\mathbf{M}$ is the weighted branch-nodes incidence matrix. As for $\mathbf{L}^B$[1], it corresponds to the Laplacian of the graph representing the network, we obtain it through the susceptance matrix $\mathbf{B}$. After replacing the power injection $\boldsymbol{p}^E$ with its equivalent $\mathbf{L}^B\boldsymbol{\theta}$. And using slack variables to transform the last set of inequality constraints into equality, the DC-OPF problem reduces to:

$$\begin{aligned} \min_{\boldsymbol{u}} \quad & F(\boldsymbol{u}) \\ \text{subject to} \quad & \underbrace{\begin{bmatrix} \mathbf{I} & \mathbf{L} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & -\mathbf{M} & \mathbf{0} & \mathbf{I} \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{z} \\ \boldsymbol{s_1} \\ \boldsymbol{s_2} \end{bmatrix}}_{\boldsymbol{u}} = \underbrace{\begin{bmatrix} \boldsymbol{d} \\ \overline{\boldsymbol{p}} \\ -\underline{\boldsymbol{p}} \end{bmatrix}}_{\tilde{\boldsymbol{d}}} \end{aligned} \tag{8.2}$$

Note that the cost only depends on the generated power. We consider here quadratic cost functions, more precisely for each *generator* agent j, we suppose $f_j(x_j) = c'_j x_j^2 + c_j x_j$ for $p^G_{j,\min} \leq x_j \leq p^G_{j,\max}$ and $+\infty$ otherwise.

Let $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_{N+2m})^T$ be the vector of Lagrangian multiplies assigned to the set of constraints in hand.

The update steps obtained for each component in $\boldsymbol{u}$ using the ADMM algorithm are given next.

With the chosen quadratic cost functions, Eq. (6.31a) that is used to update the power generation simplifies to the following:

$$x_j^{k+1} = \underset{[\underline{x_j}\,\overline{x_j}]}{\Pi} \left[\frac{\rho x_j^k - c_j - \pi_{1,j}^k - \frac{\rho}{d_1(j)} r_{1,j}^k}{2c'_j + \rho} \right]$$

where, $j \in \{1, \ldots, N\}$ and $\Pi_{[\underline{x_j}\,\overline{x_j}]}$ is the projection onto the interval $[\underline{x_j}\,\overline{x_j}]$ of the variable x_j. The residual of constraint j is

$$r_j^k = [\mathbf{A}]_j \mathbf{u}^k - d_j.$$

[1]Later we will refer to this matrix as $\mathbf{L}$ only for simplicity purpose

Let $r_{1,j} \hat{=} r_j$ be the residual of the j^{th} constraint for $j = 1, \ldots, N$. Then,

$$r_{1,j}^k = x_j^k + \sum_{i=1}^{N} L_{ji} z_i^k - d_j.$$

Let $\mathcal{N}_j$ be the set of neighbors of node j and $d_1(j) \hat{=} d(j)$ for $j = 1, \ldots, N$ be the degree of the j^{th} constraint. Then, if node j is equipped with a generator, we have

$$d_1(j) = 2 + |\mathcal{N}_j|.$$

If no generator exists at node j then the degree of the corresponding constraint is

$$d_1(j) = 1 + |\mathcal{N}_j|.$$

We also suppose $\pi_{1,j} \hat{=} \pi_j$ for $j = 1, \ldots, N$, this Lagrangian multipliers is updated using:

$$\pi_{1,j}^{k+1} = \frac{\rho}{d_1(j)} r_{1,j}^{k+1} + \pi_{1,j}^k.$$

The voltage angle at each node can be updated using:

$$\begin{aligned} z_j^{k+1} = & z_j^k - \frac{1}{\sum_{i=1}^{N} L_{ij}^2 + 2\sum_{i=1}^{m} M_{ij}^2} \Bigg(\sum_{i=1}^{N} L_{ij} \Big(\frac{\pi_{1,i}^k}{\rho} + \frac{r_{1,i}^k}{d_1(i)} \Big) \\ & + \sum_{i=1}^{m} M_{ij} \Big(\frac{\pi_{2,i}^k}{\rho} + \frac{r_{2,i}^k}{d_2(i)} \Big) - \sum_{i=1}^{m} M_{ij} \Big(\frac{\pi_{3,i}^k}{\rho} + \frac{r_{3,i}^k}{d_3(i)} \Big) \Bigg) \end{aligned}$$

where $\mathbf{L}$ is an $N \times N$ matrix, $L_{ij} = b_{ij}$ is the susceptance of line $(i, j) \in E$, and $L_{ii} = \sum_{i \sim j} - b_{ij}$. $\mathbf{M}$ is the weighted incidence matrix of size $m \times N$ where each row corresponds to an edge in E. For the k^{th} row linking nodes i and j, M_{kj} is chosen to be equal to b_{ij}, $M_{ki} = -b_{ij}$, and $M_{kj'} = 0$ if $i' \neq i$ and $i' \neq j$.

For $i = 1, \ldots, m$ we suppose $\pi_{2,i} \hat{=} \pi_{i+N}$, then $\boldsymbol{\pi}_2$ represents the second subset of $\boldsymbol{\pi}$. It is updated using

$$\pi_{2,i}^{k+1} = \frac{\rho}{d_2(i)} r_{2,i}^{k+1} + \pi_{2,i}^k$$

where $d_2(i) \hat{=} d(i + N)$ for $i = 1, \ldots, m$ is the degree of the constraints in the second part of matrix $\mathbb{A}$:

$$d_2(i) = 3.$$

$\boldsymbol{r}_2$ is used to represent the second subset of the residual vector $\boldsymbol{r}$. I.e., for $i = 1 \ldots, m$ we have $r_{2,i} \hat{=} r_{i+N}$ and

$$r_{2,i}^k = \sum_{j=1}^{N} M_{ij}\theta_j^k + s_{1,i}^k - d_{i+N}.$$

In the same manner, to the third block of rows in matrix $\mathbb{A}$ we assign the Lagrangian multipliers $\pi_{3,i} \hat{=} \pi_{i+N+m}$, for $i = 1, \ldots, m$. These multipliers are updated as follows:

$$\pi_{3,i}^{k+1} = \frac{\rho}{d_3(i)} r_{3,i}^{k+1} + \pi_{3,i}^k$$

The degree of these constraints is depicted using $d_3(i) \hat{=} d(i + N + m)$ where

$$d_3(i) = 3.$$

The corresponding residual is given by $r_{2,i} \hat{=} r_{i+N+m}$ where

$$r_{3,i}^k = \sum_{j=1}^{N} - M_{ij}\theta_j^k + s_{2,i}^k - d_{i+N+m}.$$

As for the slack variables that were used to convert the thermal limits on the transmission lines into equality constraints. Their corresponding update steps are the following.

$$\begin{aligned} s_{1,j}^{k+1} &= \max\left\{0, z_{1,j}^k - \frac{\pi_{2,j}^k}{\rho} - \frac{r_{2,j}^k}{d_2(i)}\right\} \\ s_{2,j}^{k+1} &= \max\left\{0, z_{2,j}^k - \frac{\pi_{3,j}^k}{\rho} - \frac{r_{3,j}^k}{d_3(i)}\right\}. \end{aligned}$$

In the following, we apply in different scenarios our asynchronous distributed algorithm to the DC-OPF problem using the network of the IEEE−30 and IEEE-118 bus test systems.

8.3 IEEE−30 bus test system

The IEEE−30 bus system given by Fig.8.1 contains $N = 30$ nodes and $m = 41$ transmission lines. It represents a portion of the American Electric Power System (in the Midwestern US) as of December, 1961. The nodes with symbol G are equipped with a fuel generator. The power demand, if it exists at a certain node, is designated by a bold arrow.

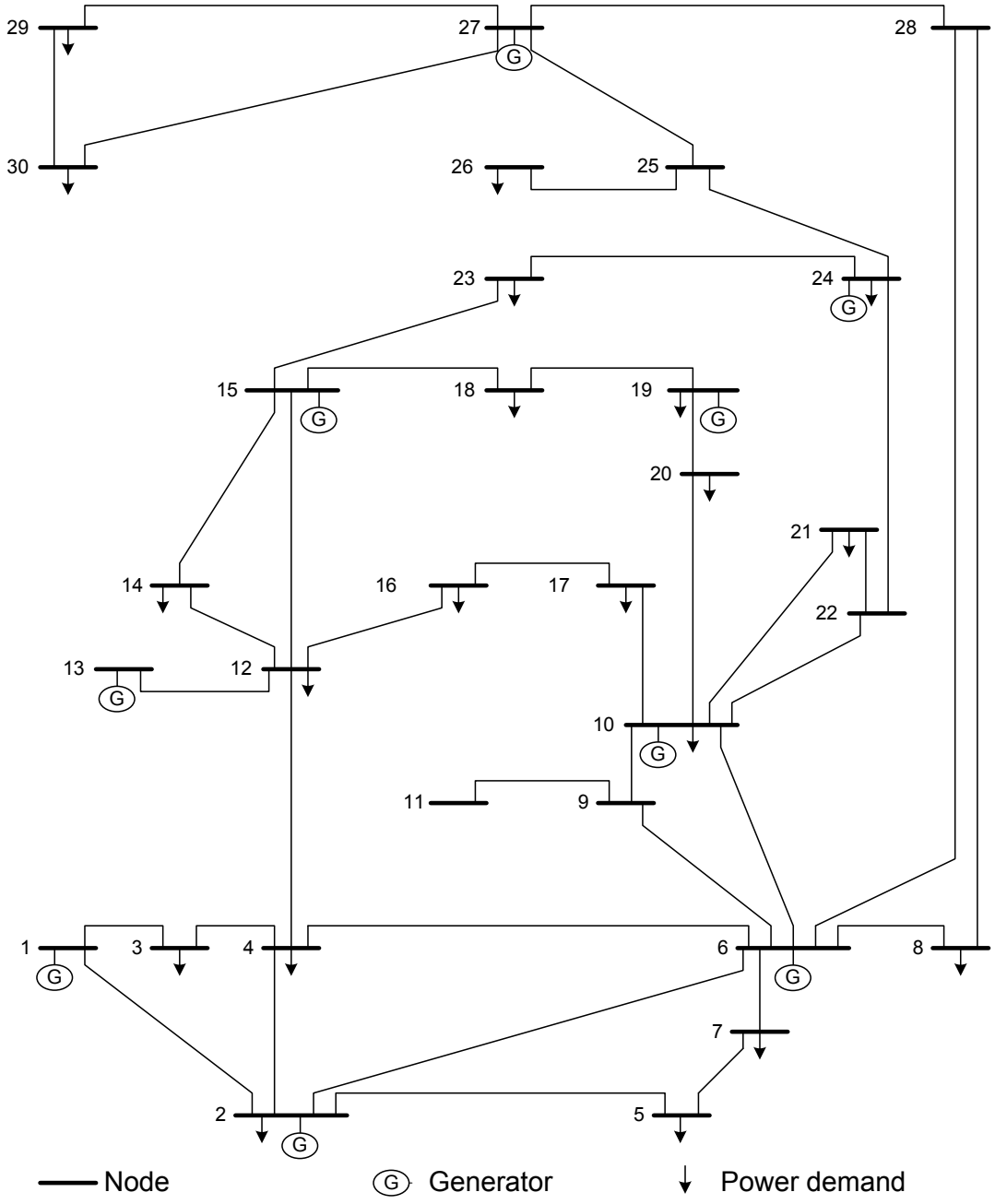

Figure 8.1: IEEE 30 bus system.

Table 8.1: Generators data, IEEE−30 Bus Test System

Node	$\underline{p_j^G}$	$\overline{p_j^G}$	c'_j	c_j
1	0	50	0.037	20
2	0	30	0.01	20
6	0	80	0.0175	10
10	0	35	0.0083	10
13	0	20	0.01	15
15	0	10	0.0625	10
19	0	20	0.01	15
24	0	10	0.0250	20
27	0	40	0.0250	20

In this simulation, we have 9 generators that are located at buses $1, 2, 6, 10, 12, 15, 19, 24,$ and 27. The data related to these generators are given in Table 8.1. The lower bound is set to zero in this case, higher values can also be used. The demand at each node is given in Table 8.2. The branches and their characteristics are provided in Table A.1.

8.3.1 Overlapping areas

Firstly, we divide the network representing the IEEE−30 bus test system into $L = 3$ overlapping areas A_1, A_2 and A_3 as per Table 8.3 and Fig. 8.2. We compare the different versions of the DPS algorithm with ADMM under different settings.

Table 8.2: Power demand data, IEEE−30 Bus Test System

Node	p_j^D	Node	p_j^D
1	0	16	3
2	22	17	9
3	5	18	3
4	8	19	9
5	15	20	2
6	0	21	17
7	12	22	0
8	20	23	3
9	0	24	8
10	5	25	0
11	0	26	3
12	10	27	0
13	0	28	0
14	14	29	2
15	0	30	10

Table 8.3: IEEE−30 bus test system division into 3 overlapping areas

Area	Nodes	Number of nodes
A_1	1-11,17,20,28	14
A_2	3,4,12-20,23	12
A_3	10,21-30	11

One area out

In this scenario, we compare the application of the synchronous ADMM DPS algorithm to the asynchronous version of the algorithm.

For the synchronous case, all the variables are updated in every iteration. This is the same basic of the centralized and fully distributed versions of the algorithm. This division between areas that synchronize their updating process is necessary in case of system with multiple authorities. When we compare based on the iteration number, the plots corresponding to the synchronous application coincides with those of the centralized and fully distributed implementations. This is mainly because we are not considering the time used for communication between nodes or areas. Thus, even if an iteration in synchronous DPS using ADMM requires a different amount of time than the other applications; this cannot be shown by comparing the results on iteration basis.

For the asynchronous case, areas A_1 and A_3 are *on* or activated together while area A_2 is activated randomly only for a fraction of the total time. First we activate A_2 for 75% of the time, then 50% and finally 25%. These are equivalent respectively to turning A_2 *off* for 25%, 50% and 75% of the total time. For each case we plot the evolution of the global cost function in Fig. 8.3. We start by checking the synchronous DPS problem. Areas A_1 and A_3 are always activated and their variables are updated at each iteration k. Area A_2 is only updated for a fraction of time. The variables of the inner nodes in A_2 are only updated when this area is *on*.

The plots in Fig. 8.3 represent the objective convergence of the corresponding algorithm. Firstly, from an implementation point of view, the synchronous DPS can be also seen as a fully distributed and even a centralized version because all the variables are being updated in every iteration. In the asynchronous scenario, only the variables related to the awakened areas are

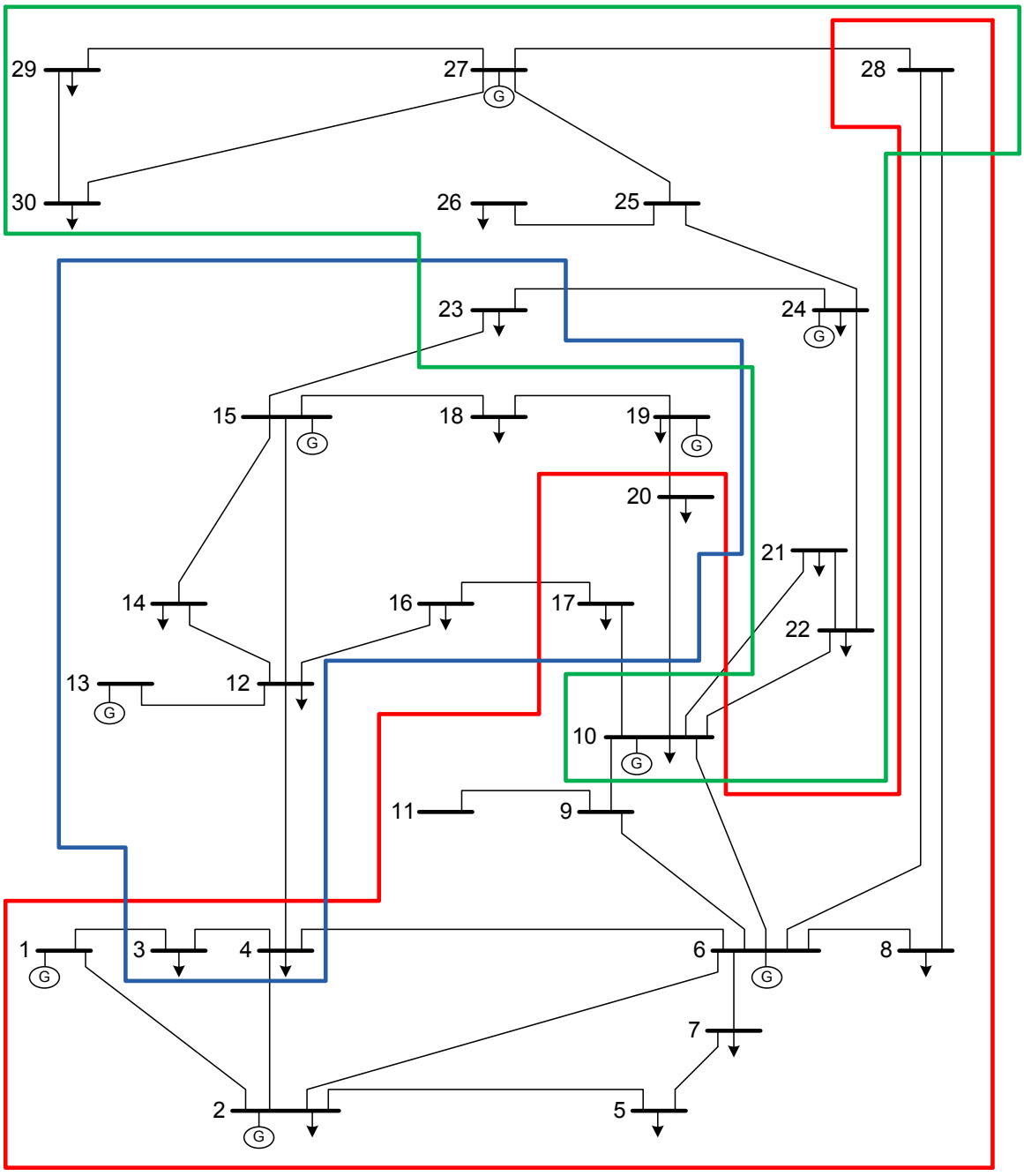

Figure 8.2: IEEE−30 bus test system divided into 3 overlapping areas.

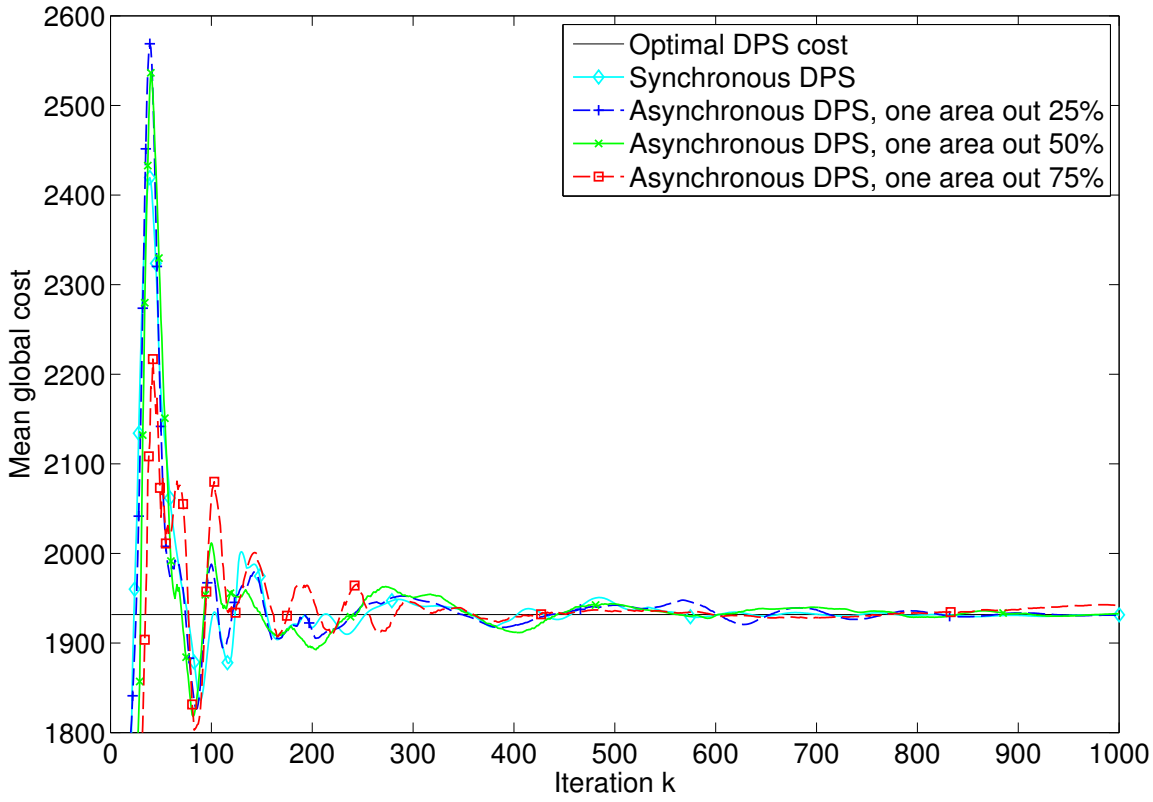

Figure 8.3: Mean global cost, overlapping areas.

going to be updated by the corresponding processor. We remark from these plots that a all the versions of the algorithm converge to the optimal cost. From the synchronous DPS plot to the asynchronous one where A_2 was *off* for 75% of the time, we can observe an increasing delay in attaining this cost. Additional iterations are required when the *off* percentage increases. Nevertheless, the increase in number of iterations is not overwhelming. Thus, when deactivating or disconnecting a part of the network for a period of time will not have a tremendous effect on the rest of the network.

After approximately 1000 iterations, the relative error percentage with respect to the optimal cost for each case are the following:

1. Synchronous DPS: 0.01%

2. Asynchronous DPS, A_2 is 25% *off*: 0.02%

3. Asynchronous DPS, A_2 is 50% *off*: 0.07%

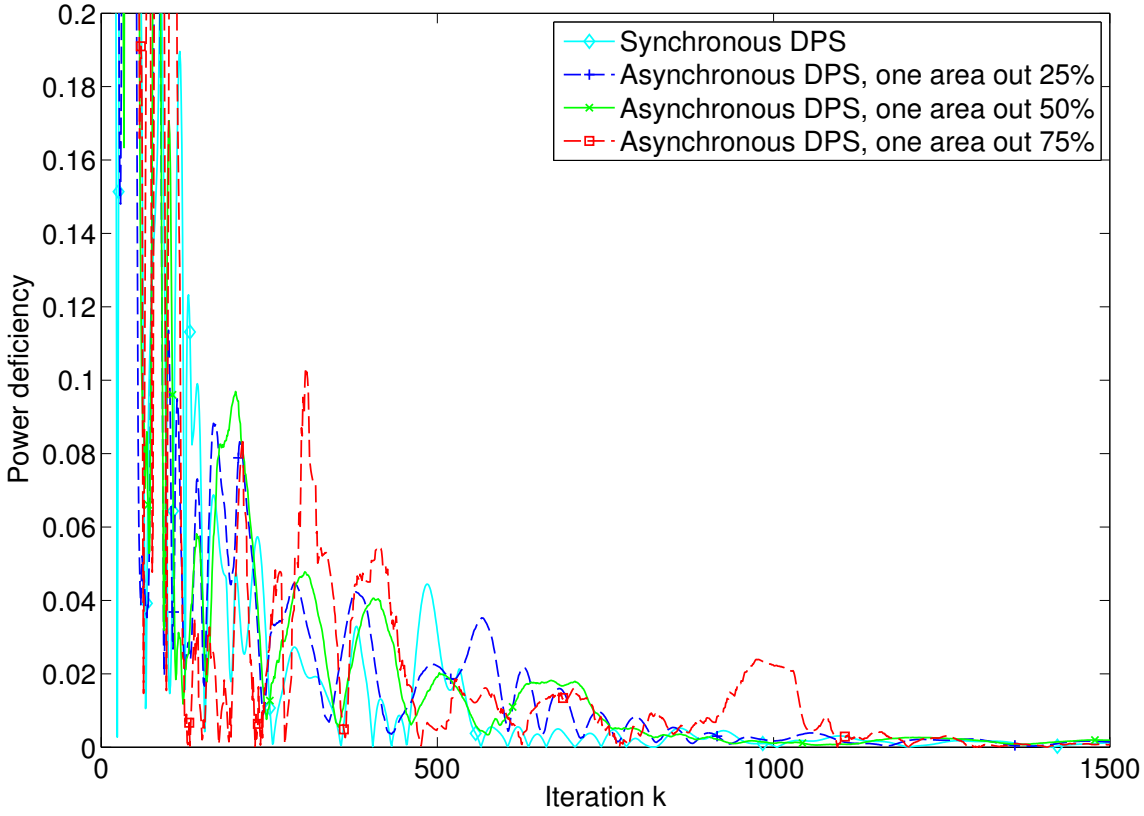

Figure 8.4: Mean global power deficiency, overlapping areas.

4. Asynchronous DPS, A_2 is 75% *off*: 0.1%.

We also check in Fig. 8.4 the sufficiency of the power demand in this network by plotting the power flow deficiency. This is equivalent to the residual of the power flow constraints. These plots show the mean global power deficiency in the network obtained by dividing the sum of all the residuals to the number of constraints in hand.

After 1000 iterations the mean global power deficiency in the network (p.u.) for each case are given below:

1. Synchronous DPS: 0.001
2. Asynchronous DPS, A_2 is 25% *off*: 0.002
3. Asynchronous DPS, A_2 is 50% *off*: 0.0024
4. Asynchronous DPS, A_2 is 75% *off*: 0.0029.

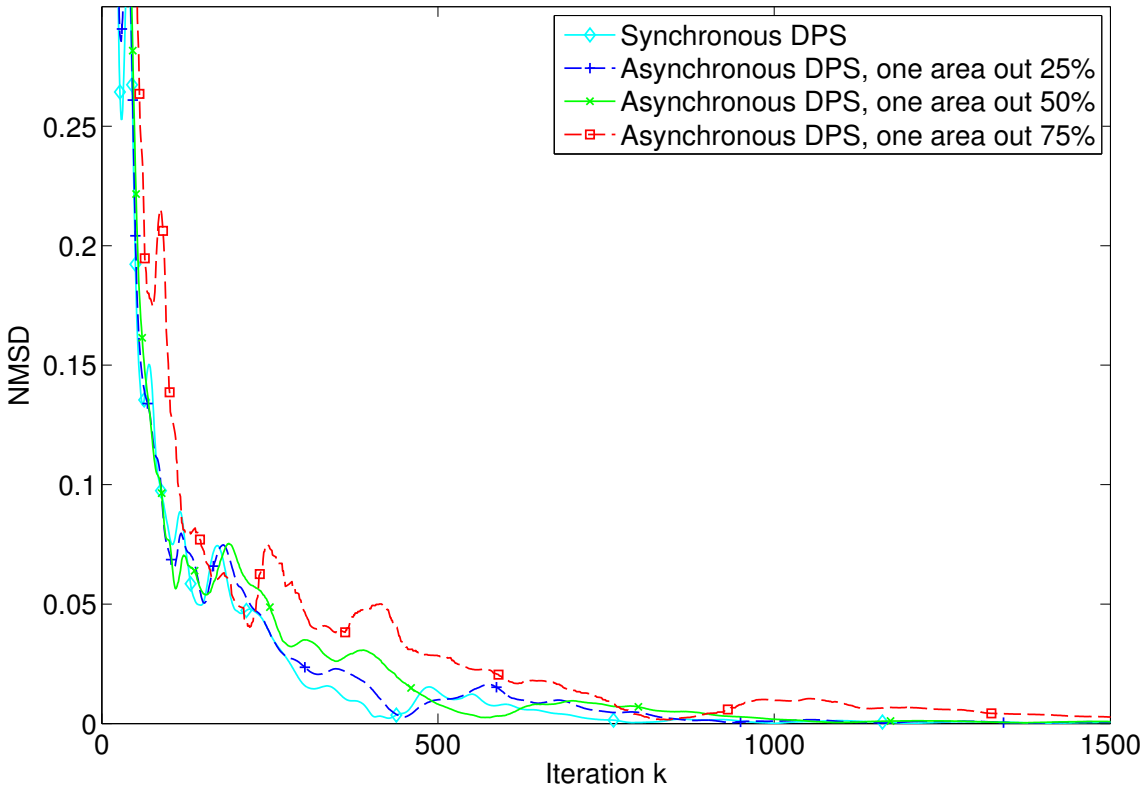

Figure 8.5: NMSD, overlapping areas.

Around 2000 iterations the residuals in all the cases are as follows:

1. Synchronous DPS: 0.0003
2. Asynchronous DPS, A_2 is 25% *off*: 0.0005
3. Asynchronous DPS, A_2 is 50% *off*: 0.0007
4. Asynchronous DPS, A_2 is 75% *off*: 0.0009.

These values along with the small relative error on the global cost prove the convergence of both the synchronous and asynchronous distributed ADMM algorithms. In order to track the error on the obtained solution, we plot the normalized mean squared deviation (NMSD) to the optimal solution versus the number of iterations. NMSD is given at the division of the squared error between the solution and the optimal values to the squared optimal value.

$$NMSD = \mathbb{E}\{\frac{\|\boldsymbol{x} - \boldsymbol{x}_{optimal}\|^2}{\|\boldsymbol{x}_{optimal}\|^2}\}.$$

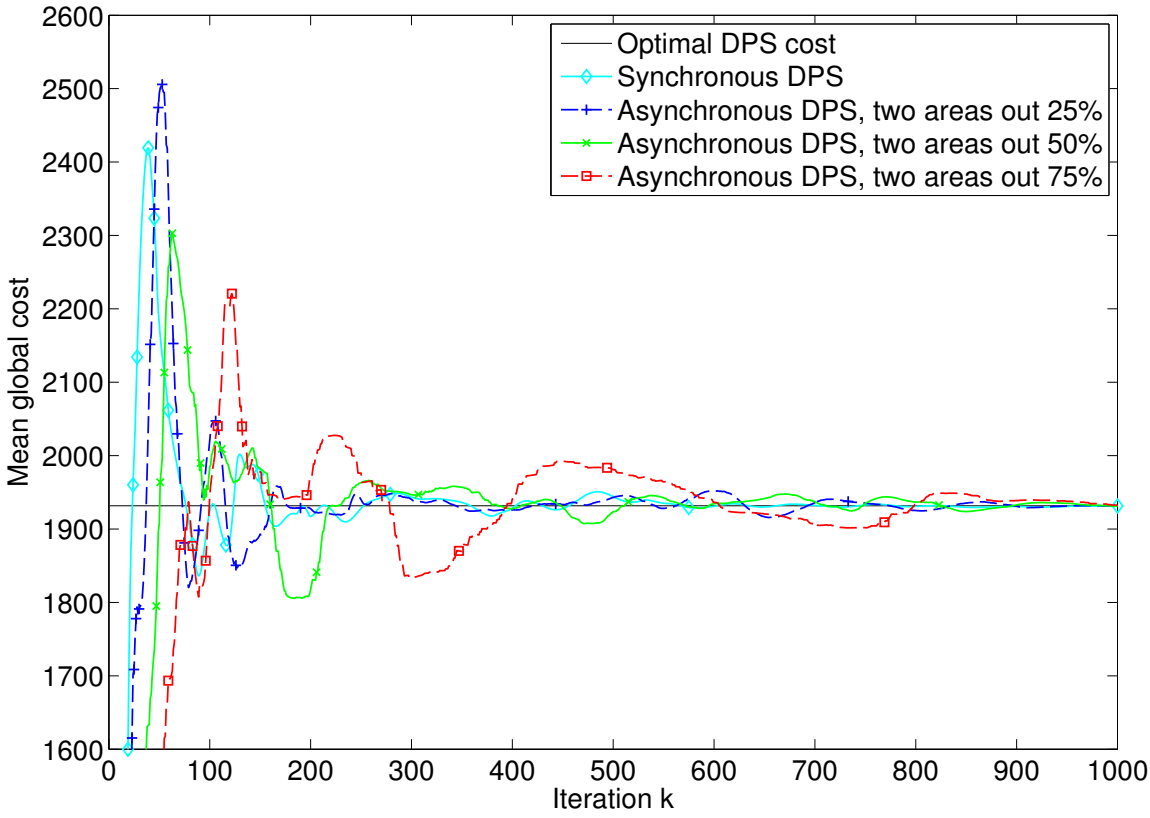

Figure 8.6: Mean global cost, overlapping areas.

The plots corresponding to the NMSD in each case are depicted in Fig. 8.5.

The NMSD for each case are as follows:

1. Synchronous DPS: 0.0001
2. Asynchronous DPS, A_2 is 25% *off*: 0.0009
3. Asynchronous DPS, A_2 is 50% *off*: 0.002
4. Asynchronous DPS, A_2 is 75% *off*: 0.009.

If we run the simulations for an additional 1000 iterations the NMSD decreases further in all the cases as given below:

1. Synchronous DPS: 0.00006
2. Asynchronous DPS, A_2 is 25% *off*: 0.0001

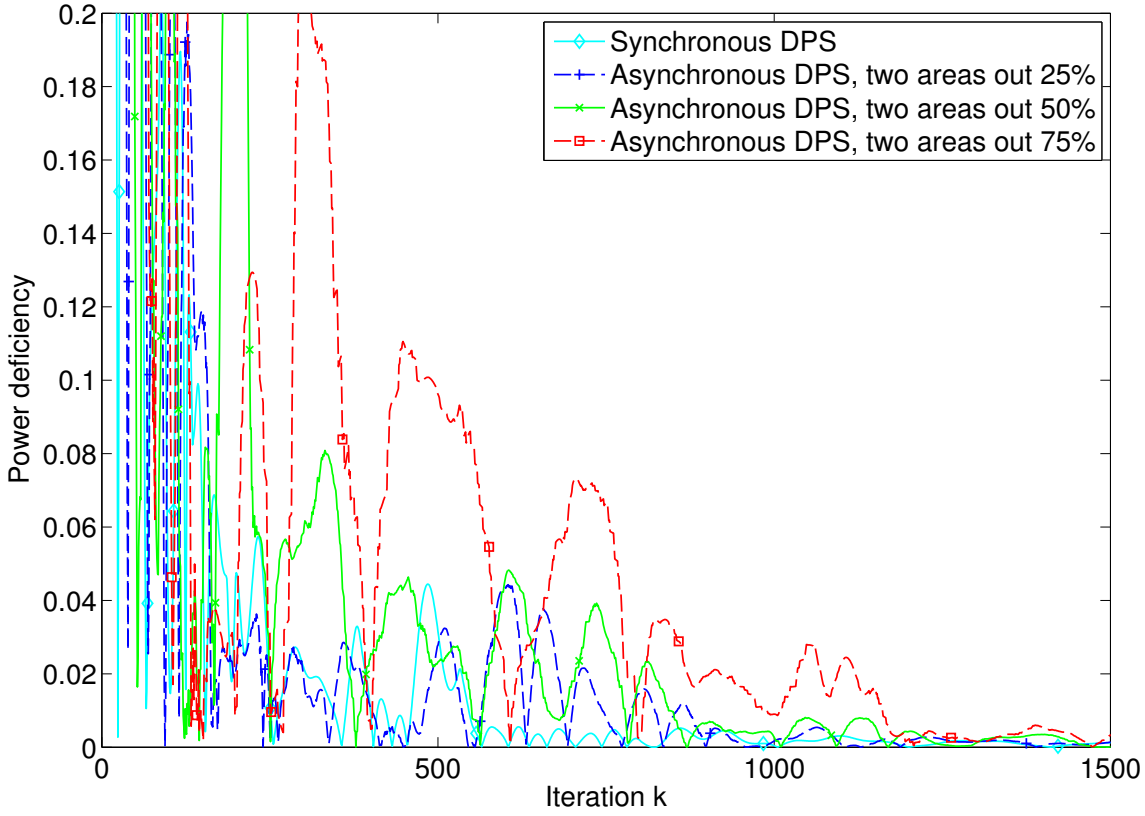

Figure 8.7: Mean global power deficiency, overlapping areas.

3. Asynchronous DPS, A_2 is 50% *off*: 0.0003

4. Asynchronous DPS, A_2 is 75% *off*: 0.0009.

Remark here that the delay from the synchronous to the asynchronous applications is well justified by the fact that while all the variables are updated in each iteration of the centralized and synchronous schemes, only a subset of the agents may be active in the asynchronous version leading to a lower computations per iteration ratio.

When observing these relatively small errors of the asynchronous ADMM application, we can see the objective and residual convergences of the ADMM method. One can conclude that even when the areas are being *activated* (or *deactivated*) randomly we can always obtain a solution that satisfies all the constraints and at a minimal cost. This would also lead to less computation and communication between the different processors and an efficient usage of the resources.

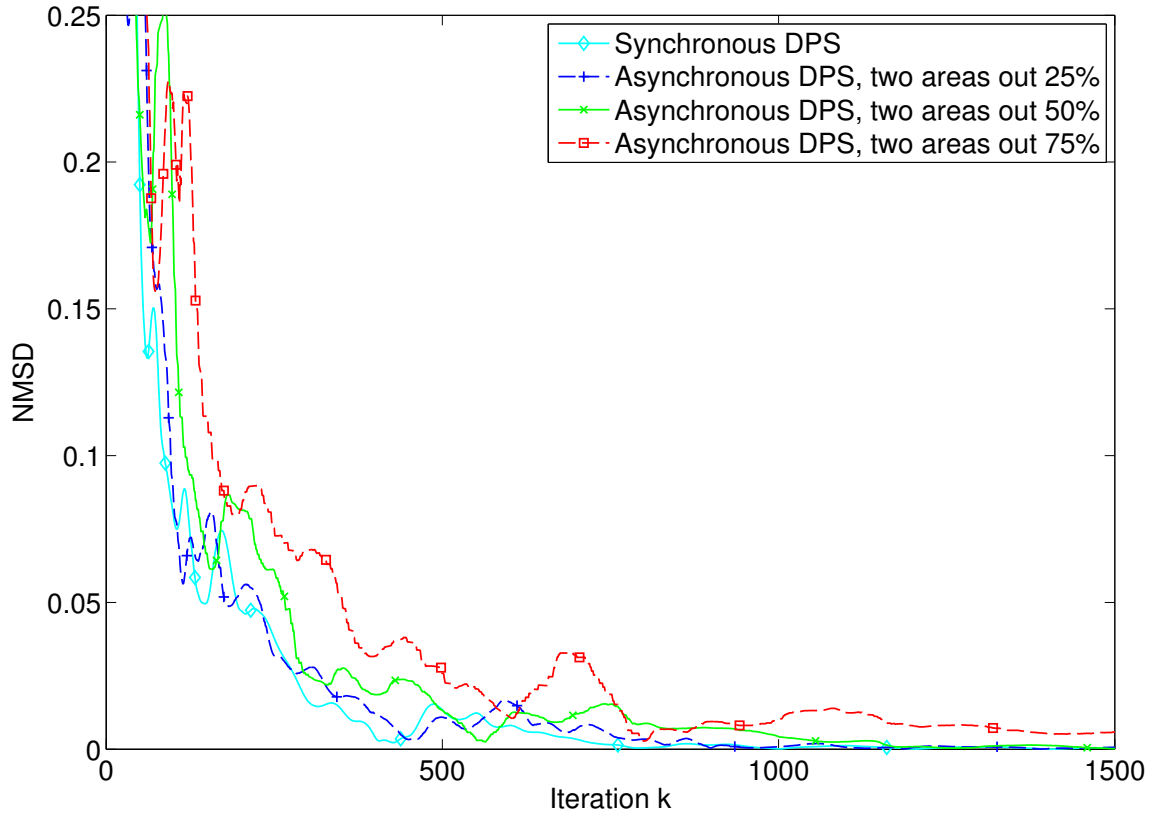

Figure 8.8: NMSD, overlapping areas.

Two areas out

In this setting, we are either activating area A_1 at certain iteration, either all the areas are being activated together. Thus, for a certain percentage of time, areas A_2 and A_3 are being turned off or not participating in the update process. Thus, A_1 is always activated while A_2 and A_3 are turned *on* only for a certain amount of time.

We start with the case where A_2 and A_3 are being activated for 75% of the time, which is equivalent to having them deactivated for 25% of the time. Then we change this percentage of time to the case where they are turned *on* and updating their variables only 50% of the time and then only for 25% (the latter is equivalent to turning them off for 75%). We plot in Fig. 8.6 the evolution of the global cost in these cases and we compare it to the synchronous implementation of the algorithm. The relative error with respect to the optimal cost for each case after approximately 1000 iterations

are as follows:

1. Synchronous DPS: 0.01%
2. Asynchronous DPS, A_2 and A_3 are 25% *off*: 0.03%
3. Asynchronous DPS, A_2 and A_3 are 50% *off*: 0.08%
4. Asynchronous DPS, A_2 and A_3 are 75% *off*: 0.1%.

The primal residuals for each iteration and scenario are plotted in Fig. 8.7. This primal residual can also be interpreted as the deficiency regarding the power demand in the network and the amount of existing shedding. After 1000 iterations the mean global power deficiency in the network (p.u.) for each case are given below:

1. Synchronous DPS: 0.001
2. Asynchronous DPS, A_2 and A_3 are 25% *off*: 0.0022
3. Asynchronous DPS, A_2 and A_3 are 50% *off*: 0.003
4. Asynchronous DPS, A_2 and A_3 are 75% *off*: 0.005.

Again, we track the error on the primal variables values by checking the NMSD to the optimal solution with respect to the iteration number. The respective plots for each case are depicted in Fig. 8.8.

The NMSD for each case are as follows:

1. Synchronous DPS: 0.0001
2. Asynchronous DPS, A_2 and A_3 are 25% *off*: 0.00095
3. Asynchronous DPS, A_2 and A_3 are 50% *off*: 0.004
4. Asynchronous DPS, A_2 and A_3 are 75% *off*: 0.01.

Nodes failure

We return our attention to the whole undivided network. We test the application of the algorithms in the case of nodes failure. We assume that cascades in the grid are triggered by random node failure events and that nodes fail independently of each other. That is, the whole network except for random nodes is being updated every iteration. In this scenario, nodes $1-4, 27$ are randomly switched off for 50% of the time and nodes $10-14, 19, 22, 28-30$

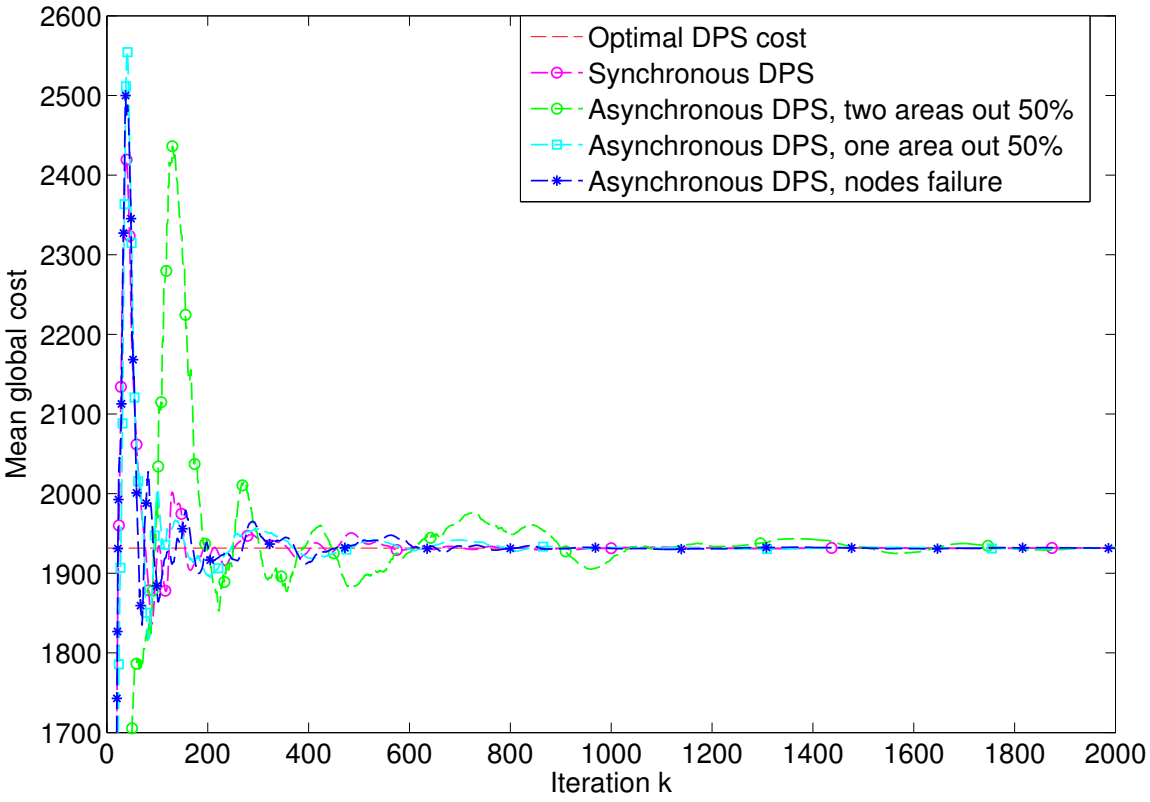

Figure 8.9: Mean global cost, overlapping areas.

are also randomly switched off for 50% of the time. As an example, in a certain iteration when nodes $1-4$ are deactivated, the rest of the network is considered as one area that is currently activated. In another iteration, when nodes $28-30$ are deactivated, the rest of the network, represented by another area, is then activated.

We compare in Fig. 8.9 the convergence of the cost in this case with the cost when one or two areas are randomly activated in each iteration for 50% of the time.

The relative error with respect to the optimal cost for each case after approximately 2000 iterations are as follows:

1. Synchronous DPS: 0.003%

2. Asynchronous DPS, nodes failure: 0.01%

3. Asynchronous DPS, A_2 is 50% *off*: 0.012%

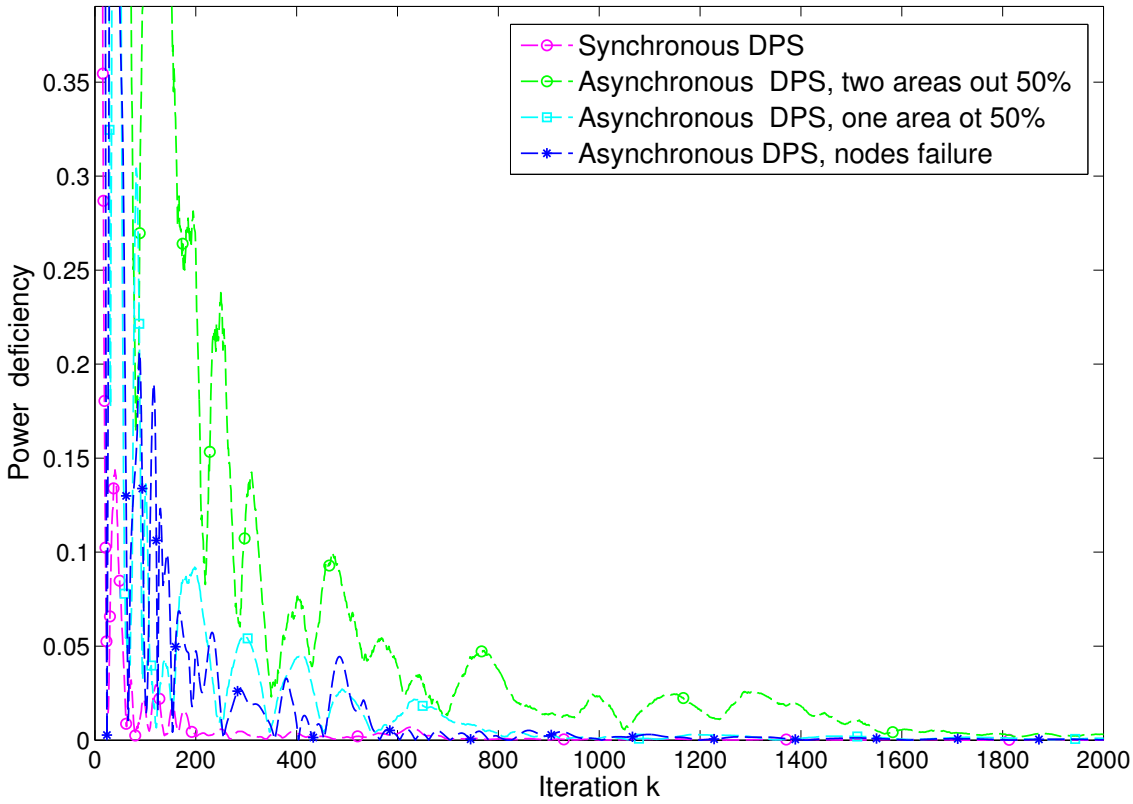

Figure 8.10: Mean global power deficiency, overlapping areas.

4. Asynchronous DPS, A_2 and A_3 are 50% *off*: 0.05%.

The power deficiency for each case are provided in Fig. 8.10.After 2000 iterations the mean global power deficiency in the network (p.u.) for each case are given below:

1. Synchronous DPS: 0.00004

2. Asynchronous DPS, nodes failure: 0.00014

3. Asynchronous DPS, A_2 is 50% *off*: 0.001

4. Asynchronous DPS, A_2 and A_3 are 50% *off*: 0.003.

Remark here that while every agent is active in the centralized scheme, only a subset of the agents may be active in the asynchronous version leading to a lower computations per iteration ratio.

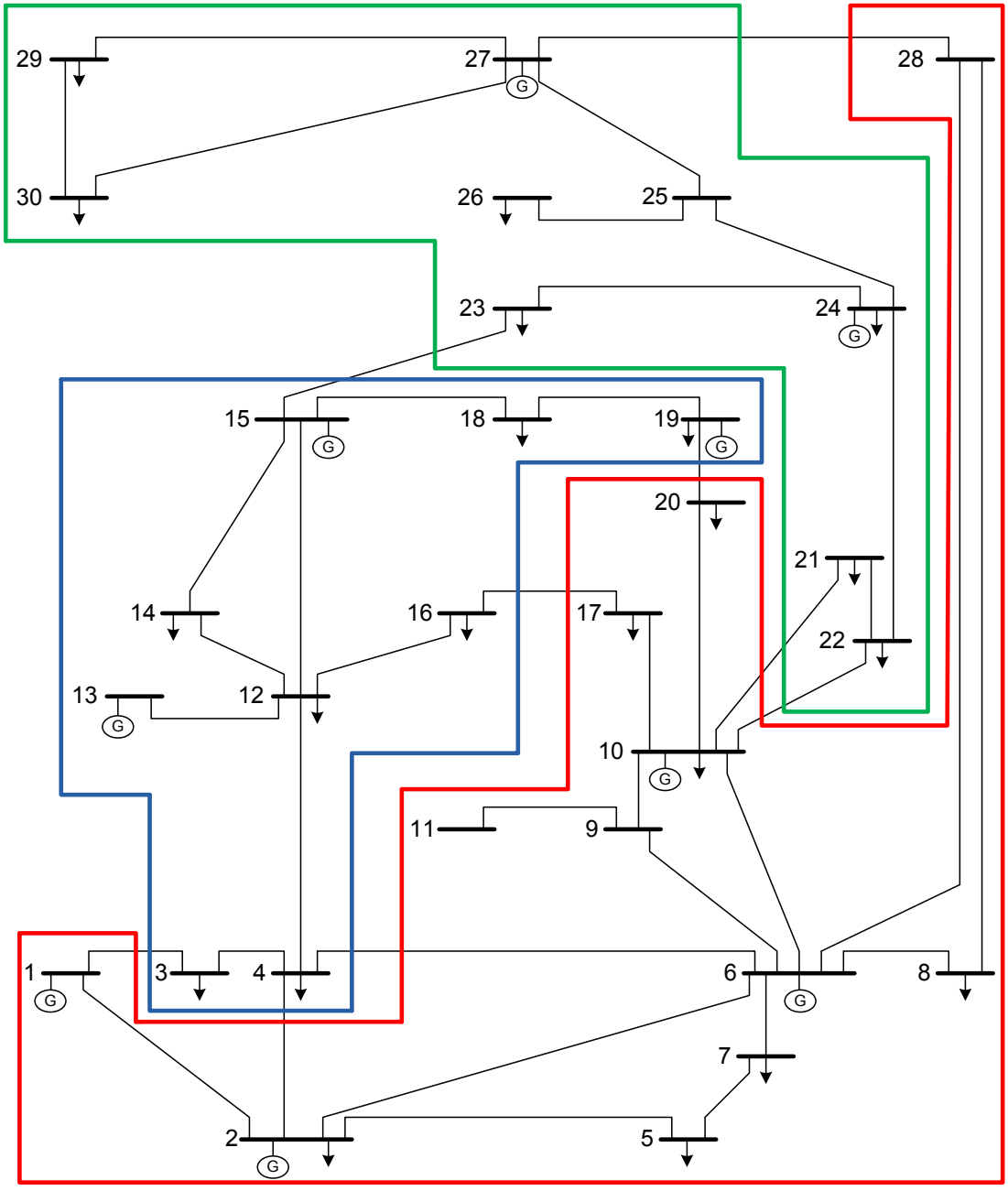

Figure 8.11: IEEE−30 bus test system divided into 3 non-overlapping areas.

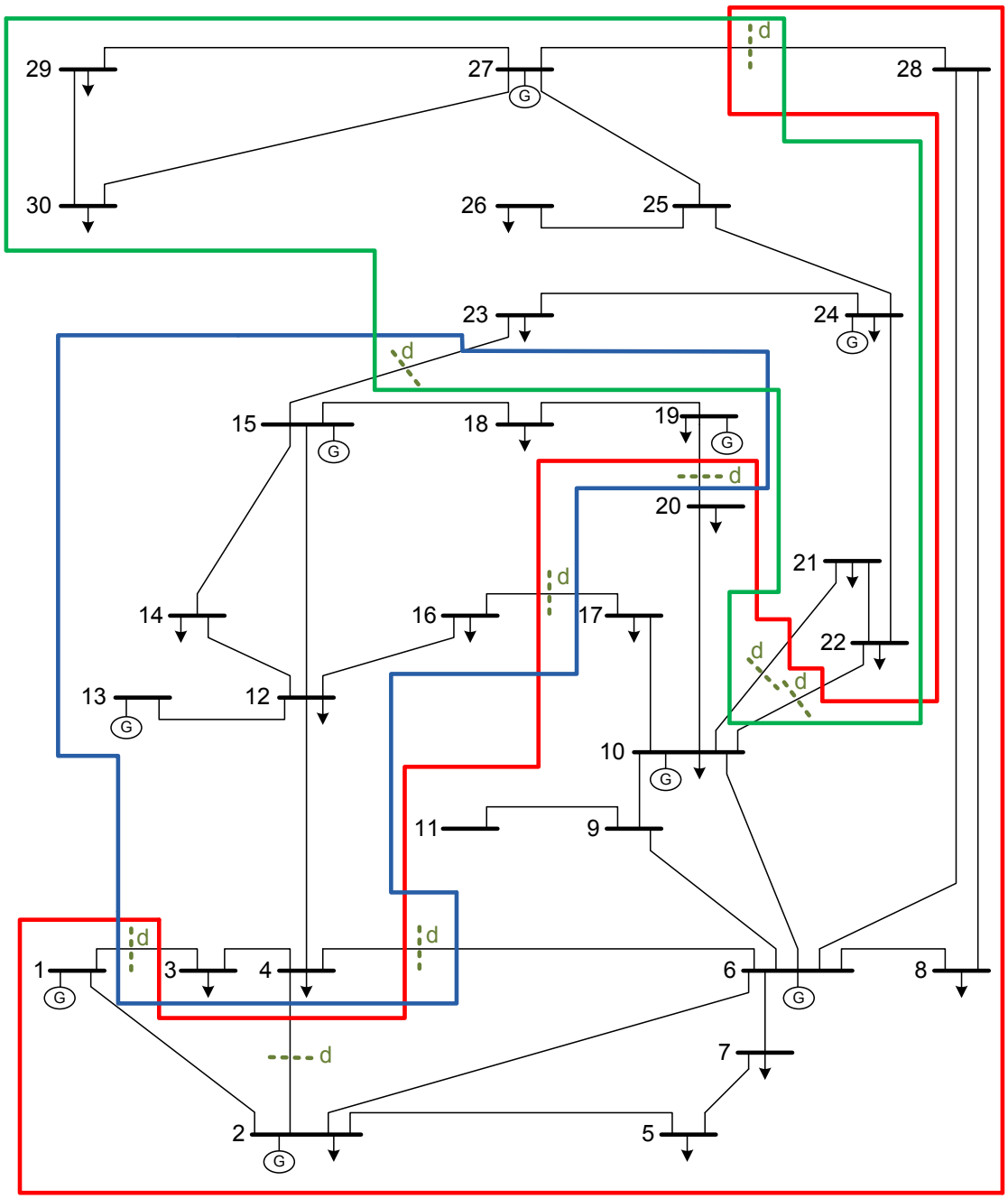

Figure 8.12: IEEE−30 bus test system divided into 3 non-overlapping areas.

Table 8.4: IEEE−30 bus test system division into 3 non-overlapping areas

Area	Nodes	Number of nodes
A_1	1,2,5-11,17,20,28	12
A_2	3,4,12-19	9
A_3	21-27,29,30	9

8.3.2 Non-overlapping areas

We consider the same IEEE−30 bus system and we divide it into $L = 3$ *non-overlapping* areas as given by Fig. 8.11 and Table 8.4.

As explained in Section 6.5, the synchronous and asynchronous distributed ADMM algorithms can be applied to this type of architecture by use of dummy nodes. Thus, we modify this network by introducing 9 dummy nodes on the tie-lines linking two different areas as given by Fig. 8.12. We obtain a new network of 39 nodes and 50 branches. The dummy nodes do not interfere in the optimization problem. The nodes do not have ability to produce or consume power. They only act as a relay to share power between the different areas. They do not generate power and no cost exist on their variables.

Next, after modifying the values of the Laplacian matrix according to the results given in Section 6.5, we apply the synchronous and asynchronous area-based distributed ADMM to the modified network using scenarios similar to those implemented in the previous section.

One area out

When dividing the areas in a non-overlapping architecture, these areas are linked only via the tie-lines connecting border nodes of the different areas. Thus, these lines are the only way to share production between the areas.

In this setting and within a certain iteration k, areas A_1 and A_3 are activated together while A_2 is switched *off* and vice versa. We can see this as a situation where A_1 and A_3 are joined into a bigger area A_4 and A_2 acts solely. In the other case, the implementation is the inverse. That is, areas A_1 and A_3 (or just A_4) are switched off together while A_2 is activated. The variables belonging to the nodes within the chosen area are the only ones to

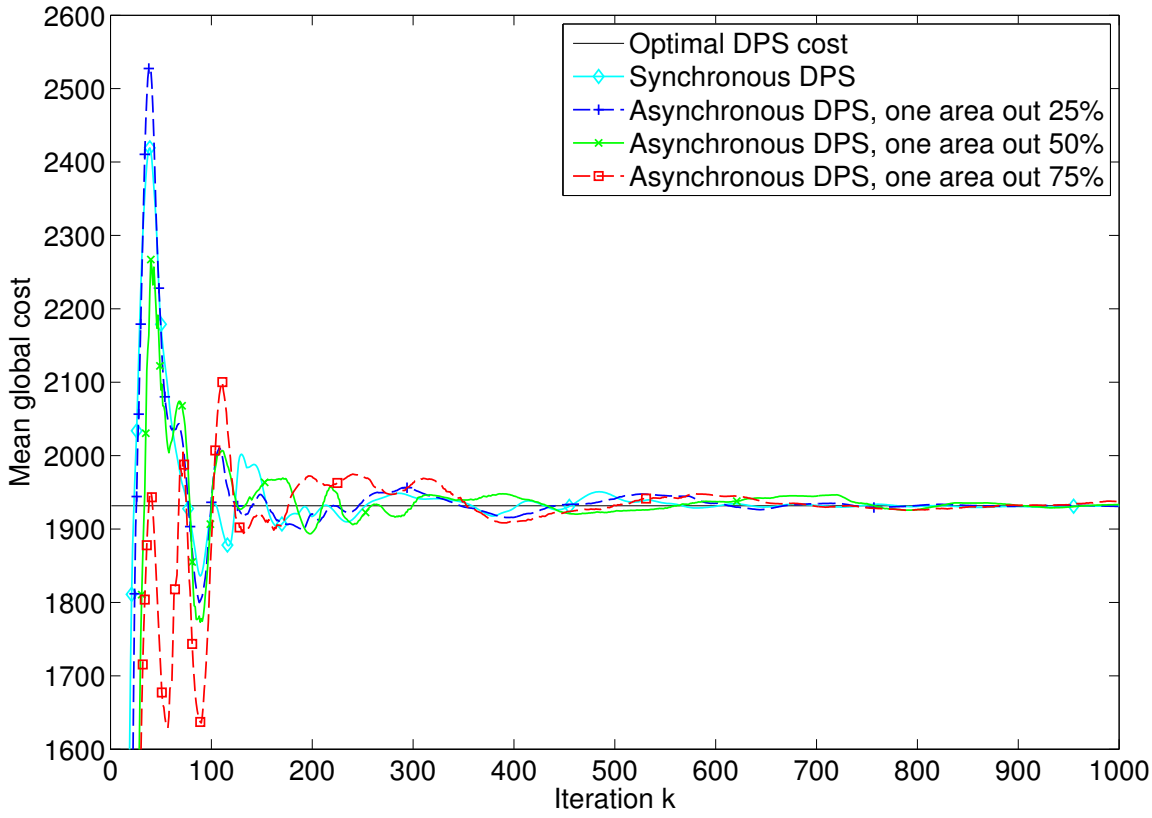

Figure 8.13: Mean global cost, non-overlapping areas.

be updated in a given iteration. Next, a communication step through the shared nodes or between the processors is performed so that these processors can update the Lagrangian multipliers.

As noted, the areas do not overlap, and from an iteration to the following, the sub-network obtained also do not overlap with the preceding sub-network. Each area is activated for a given percentage of the total time. The plots corresponding to the mean global cost are provided in Fig. 8.13. These results demonstrate the objective convergence of the corresponding algorithm.

The relative error with respect to the optimal cost for each case after approximately 1000 iterations are as follows:

1. Synchronous DPS: 0.011%

2. Asynchronous DPS, A_2 and A_3 are 25% *off*: 0.03%

3. Asynchronous DPS,A_2 and A_3 are 50% *off*: 0.085%

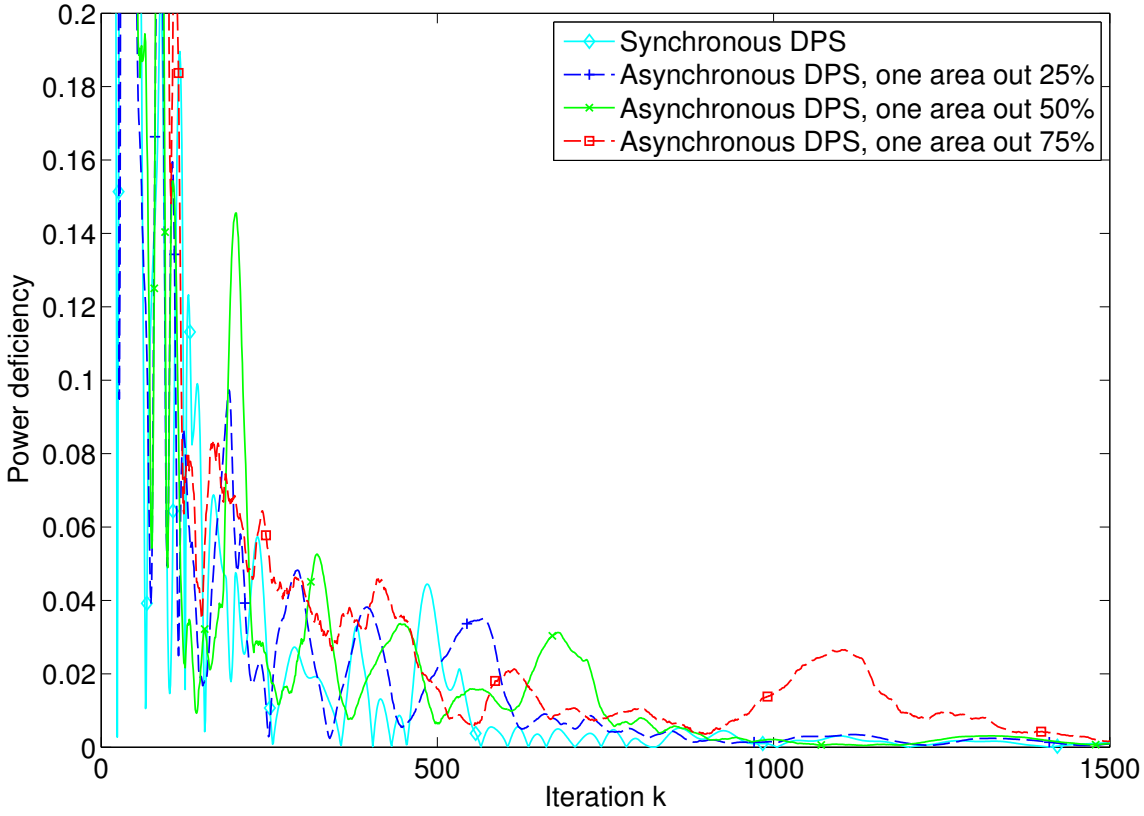

Figure 8.14: Mean global power deficiency, non-overlapping areas.

4. Asynchronous DPS, A_2 and A_3 are 75% *off*: 0.2%.

Additionally, after approximately 1400 iterations they decreases further beyond these values:

1. Synchronous DPS: 0.003%
2. Asynchronous DPS, A_2 and A_3 are 25% *off*: 0.01%
3. Asynchronous DPS,A_2 and A_3 are 50% *off*: 0.015%
4. Asynchronous DPS, A_2 and A_3 are 75% *off*: 0.02%.

We track in Fig. 8.14 the sufficiency of the power demand in this network by plotting the of the power flow constraints. These plots show the mean global power deficiency in the network obtained by dividing the sum of all the residuals to the number of constraints in hand. In 1400 iterations the mean global power deficiency in the network (p.u.) for each case are as follows:

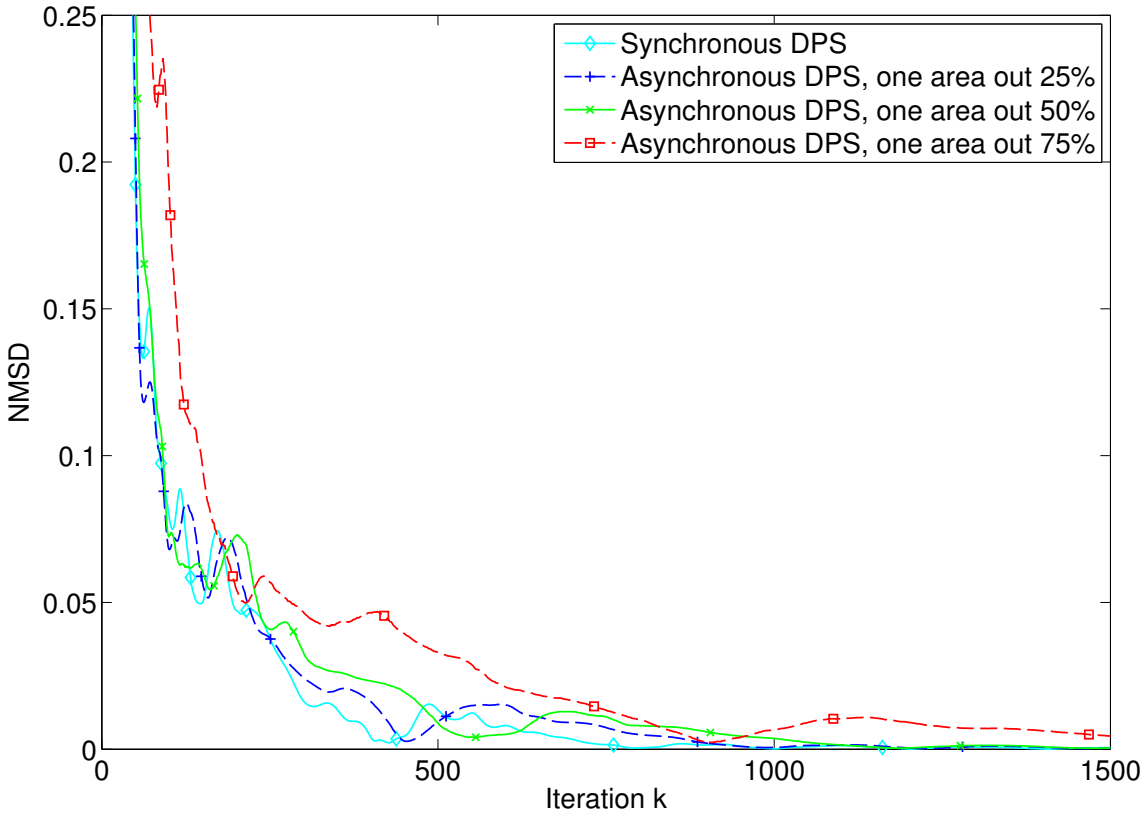

Figure 8.15: NMSD, non-overlapping areas.

1. Synchronous DPS: 0.0012
2. Asynchronous DPS, nodes failure: 0.0022
3. Asynchronous DPS, A_2 is 50% *off*: 0.0028
4. Asynchronous DPS, A_2 and A_3 are 50% *off*: 0.0042.

After increasing the number of iterations to 3000 iterations, the mean global power deficiency decreases beyond 10^{-5} p.u. for all the cases. The plots of the synchronous and asynchronous ADMM will approximately overlap at this point.

The error on the primal values depicted as NMSD can be observed using the plots of Fig. 8.15. The values corresponding to the primal variables converge to the optimal values, after 1400 iterations the NMSD for each case are given below:

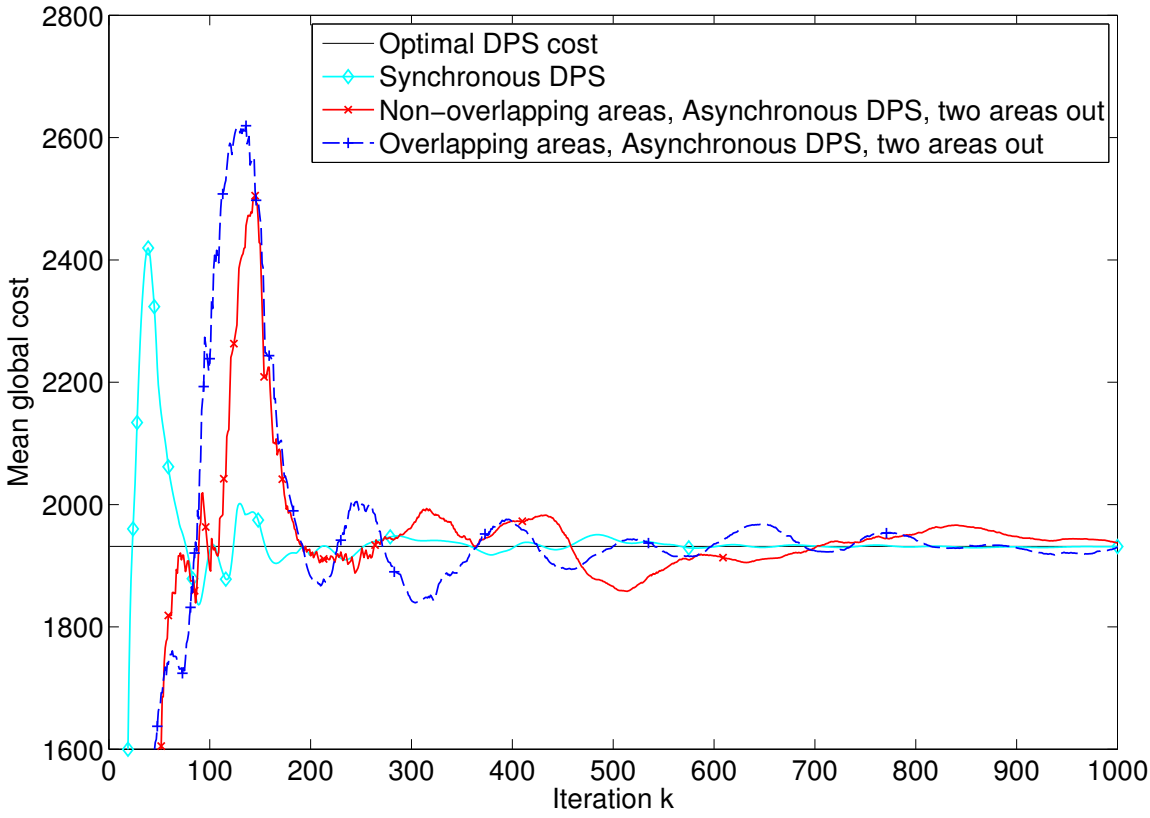

Figure 8.16: Mean global cost, non-overlapping and overlapping areas.

1. Synchronous DPS: 0.000012
2. Asynchronous DPS, nodes failure: 0.0006
3. Asynchronous DPS, A_2 is 50% *off*: 0.0009
4. Asynchronous DPS, A_2 and A_3 are 50% *off*: 0.003.

It is certain that if the number of nodes failing increases, then the convergence will be delayed by a certain number of iterations. As we can see when comparing the case of one area and two areas off.

Two areas out

We consider a different scenario where each area is only activated for 33% percent of the time. At a certain iteration k, it is either A_1 or A_2 or A_3 that

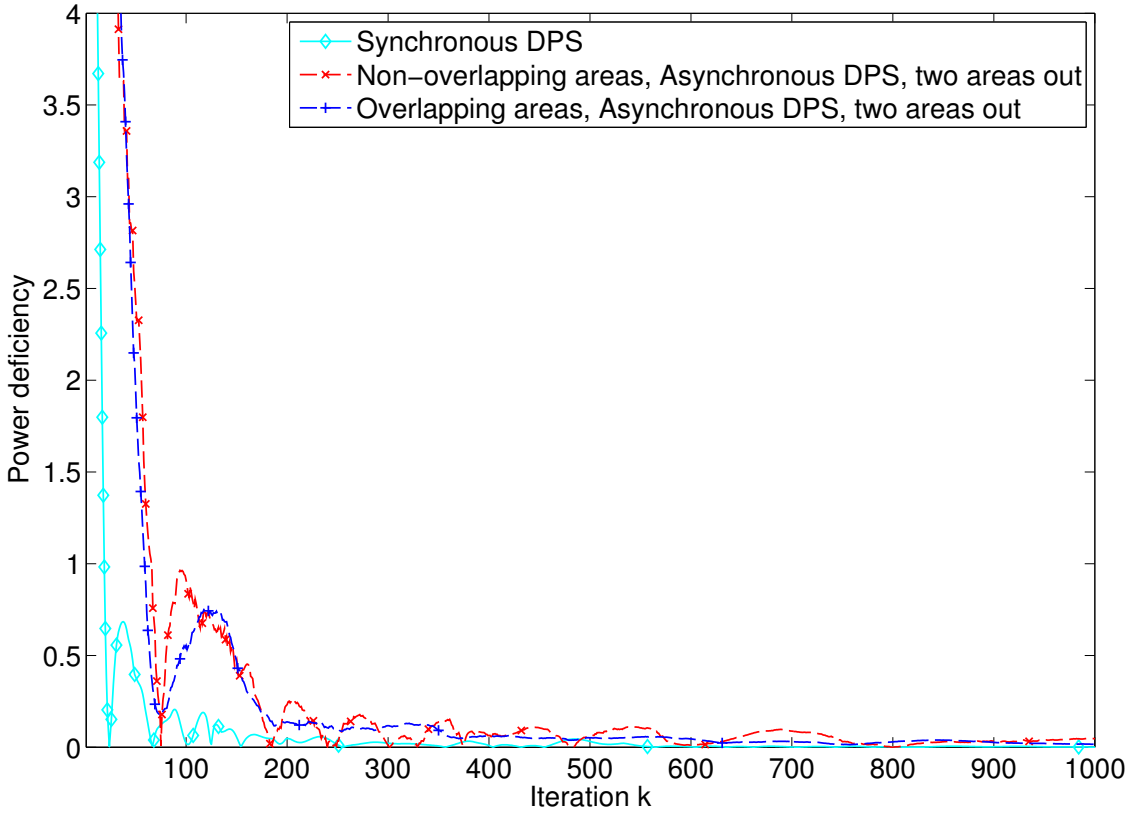

Figure 8.17: Mean global power deficiency, non-overlapping and overlapping areas.

is switched on. We compare the asynchronous DPS with ADMM for the non-overlapping and overlapping areas. The results are depicted in Fig. 8.16 that shows the evolution of the mean global cost, Fig. 8.17 for the mean global power deficiency in the network and Fig. 8.18 that depicts the NMSD of the primal variables to the optimal value.

As observed from the plots, the convergence is slightly slower when the areas do not overlap. This is mainly because the variables of the shared nodes are being updated more frequently in the overlapping case.

Thus, even in the case of a network with multiple authorities, an agreement on the minimum of the global cost can be met using the distributed production sharing w/ ADMM algorithm. The convergence of the synchronous or asynchronous distributed ADMM is always guaranteed.

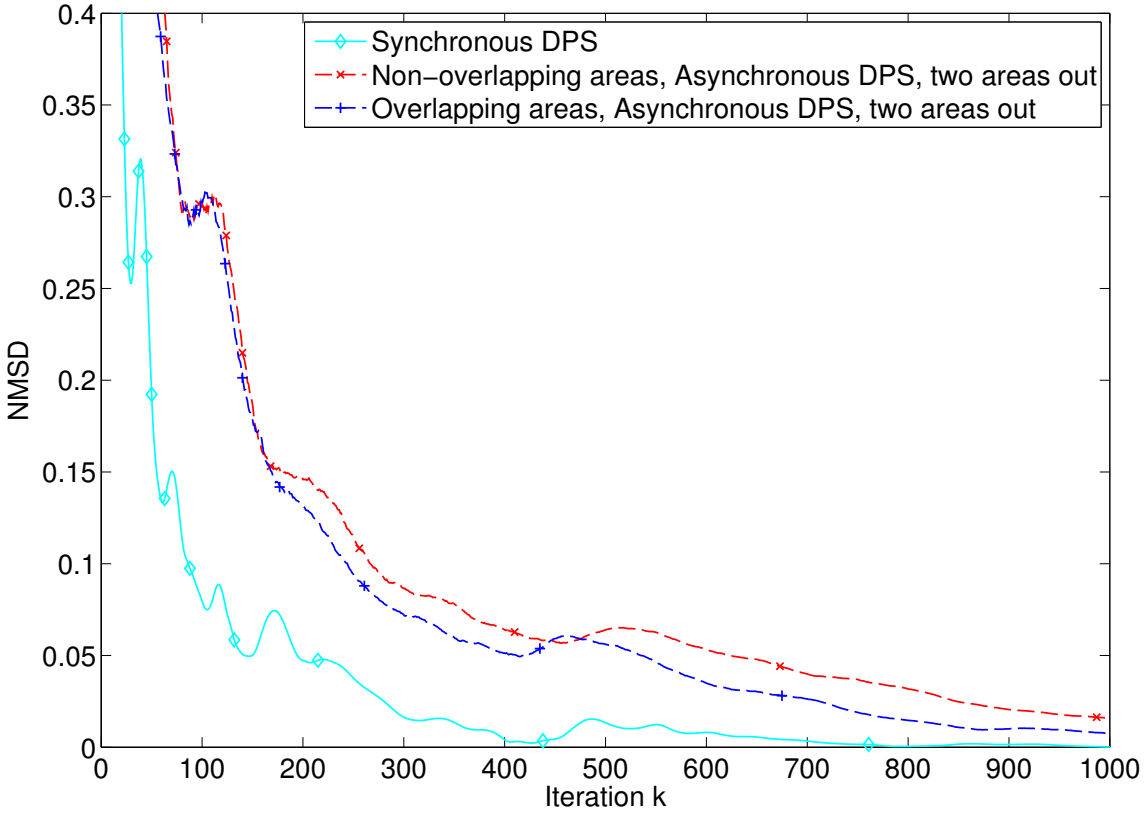

Figure 8.18: NMSD, non-overlapping and overlapping areas.

Table 8.5: IEEE−118 bus test system division into 3 overlapping areas

Area	Nodes	Number of nodes
A_1	1-34,38,113-115,117	39
A_2	24,33-75,116	45
A_3	68,69,75-112,118	41

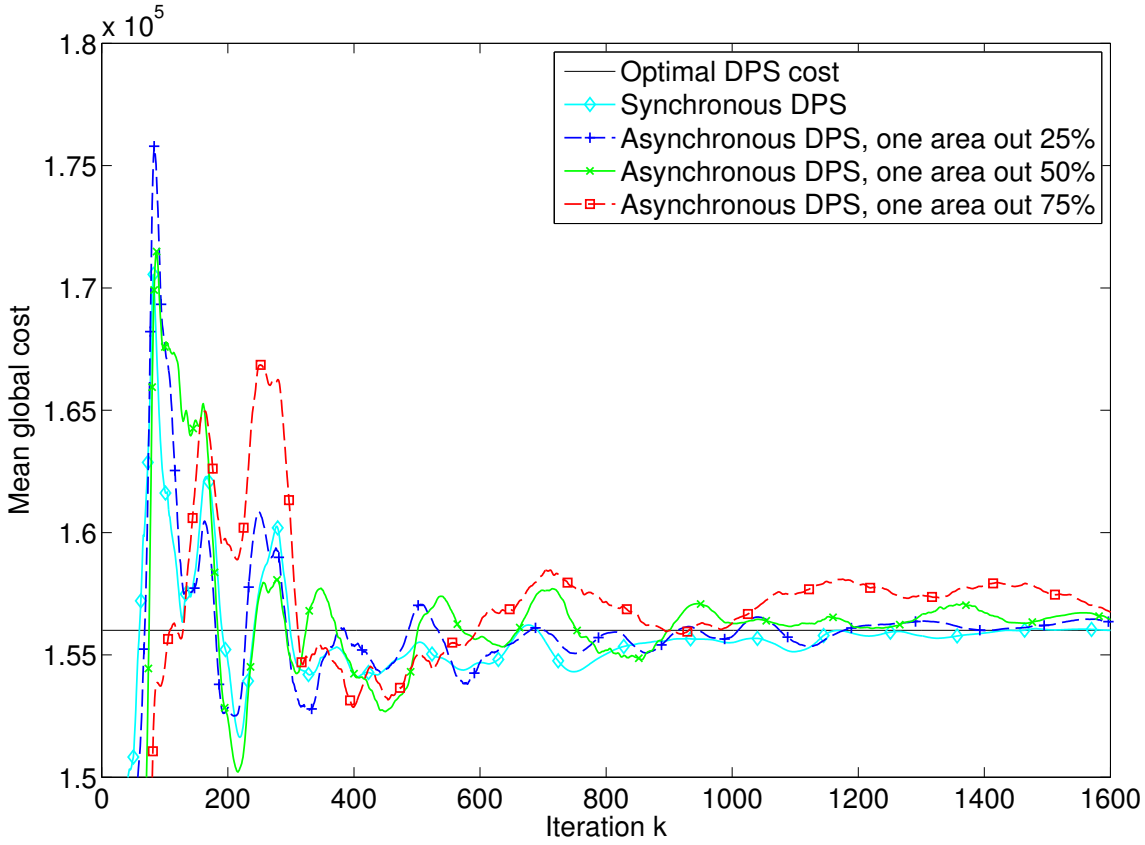

Figure 8.19: Mean global cost, IEEE−118 Bus system.

8.4 IEEE−118 bus test system

We test our algorithms with larger areas using the IEEE−118 bus test system [97] as our network. This network consists of $N = 118$ buses, 54 generators and 186 branches. The data related to the demand and generation for each node are given in Table A.2. We divide the graph representing the network into 3 overlapping areas as in Table 8.5. The transmission lines characteristics used in this implementation are provided in [97]. First we start by checking the optimal cost for the synchronous area-based distributed ADMM algorithm. Then, we continue with implementing different scenarios for the asynchronous distributed ADMM algorithm as follows.

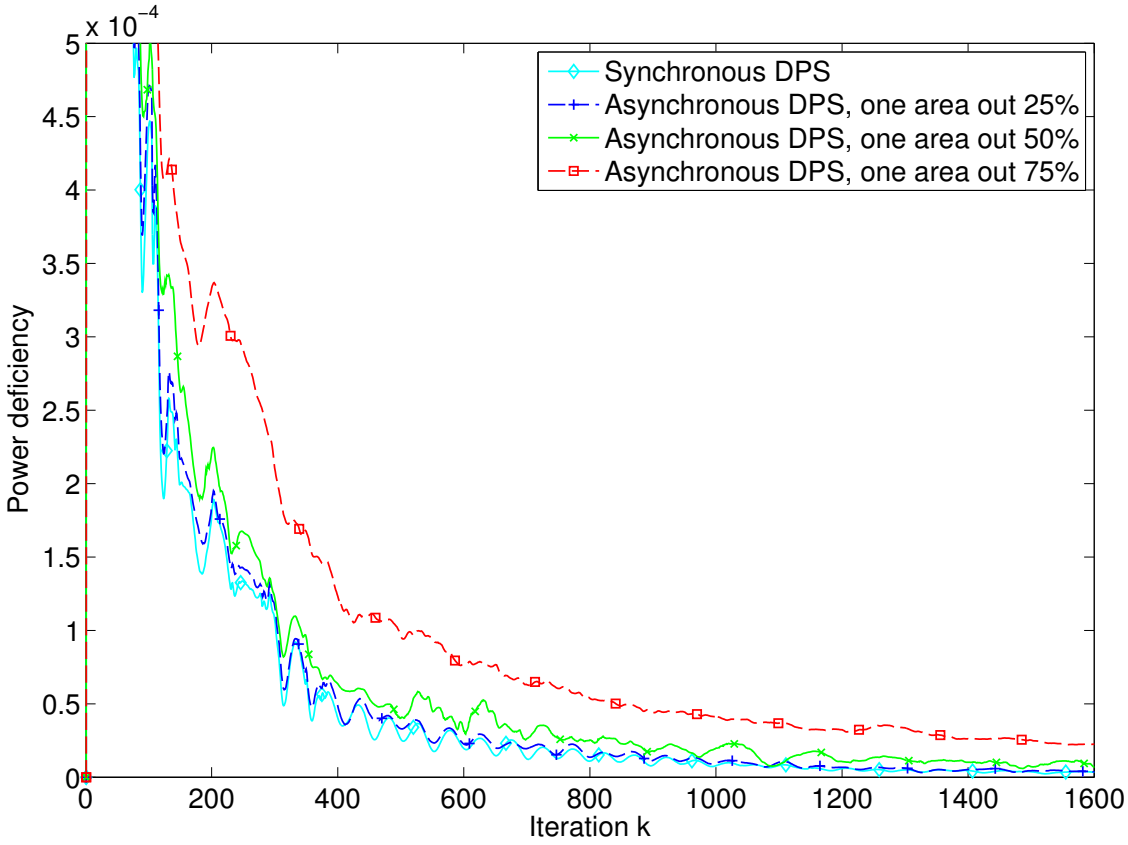

Figure 8.20: Mean global power deficiency, IEEE−118 Bus system.

One area out

As in the previous sections, we start with the case where two areas, A_1 and A_3, update their variables every iteration while area A_2 is only activated 75% of the time. Thus, A_2 is switched *off* for 25% of the time, which gives us the plot of one area 25% out. In another time, we reduce this *on* time of A_2 to 50%. Thus, A_2 is now updating its variables only half the time and it is switched off for the second half randomly. In the last part, we decrease the *on* time of A_2 further. It is considered to be awaken only for 25% of the time on random basis, this leads to the plot where one area is out 75% of the time.

The objective convergence can be seen from the plots provided in Fig. 8.19.

After approximately 1600 iterations, the relative error percentage with respect to the optimal cost for each case are the following:

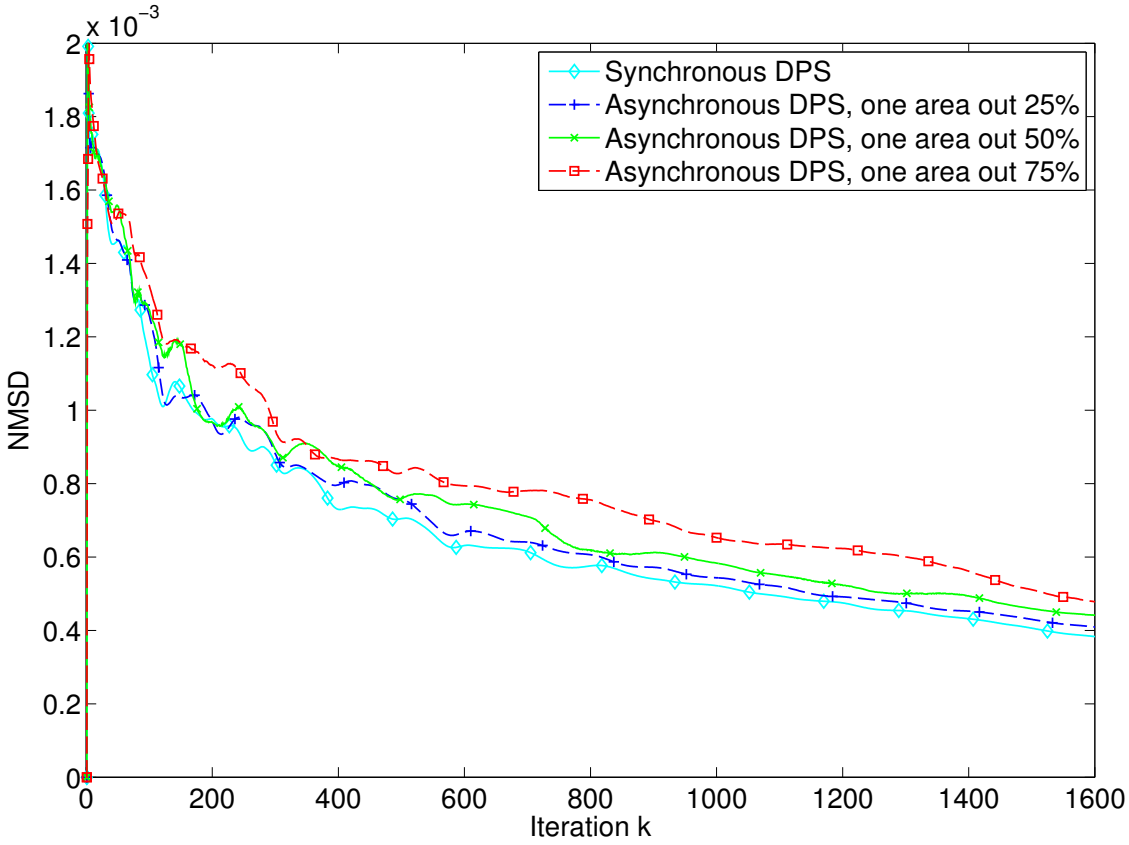

Figure 8.21: NMSD, IEEE−118 Bus system.

1. Synchronous DPS: 0.01%
2. Asynchronous DPS, A_2 is 25% *off*: 0.2%
3. Asynchronous DPS, A_2 is 50% *off*: 0.3%
4. Asynchronous DPS, A_2 is 75% *off*: 0.45%.

The primal residual convergence can be observed in Fig. 8.20. The plots track the evolution of the mean global power deficiency in the network.

Around 1600 iterations the residuals in all the cases are as follows:

1. Synchronous DPS: 3×10^{-6}
2. Asynchronous DPS, A_2 is 25% *off*: 3.2×10^{-6}
3. Asynchronous DPS, A_2 is 50% *off*: 6×10^{-6}

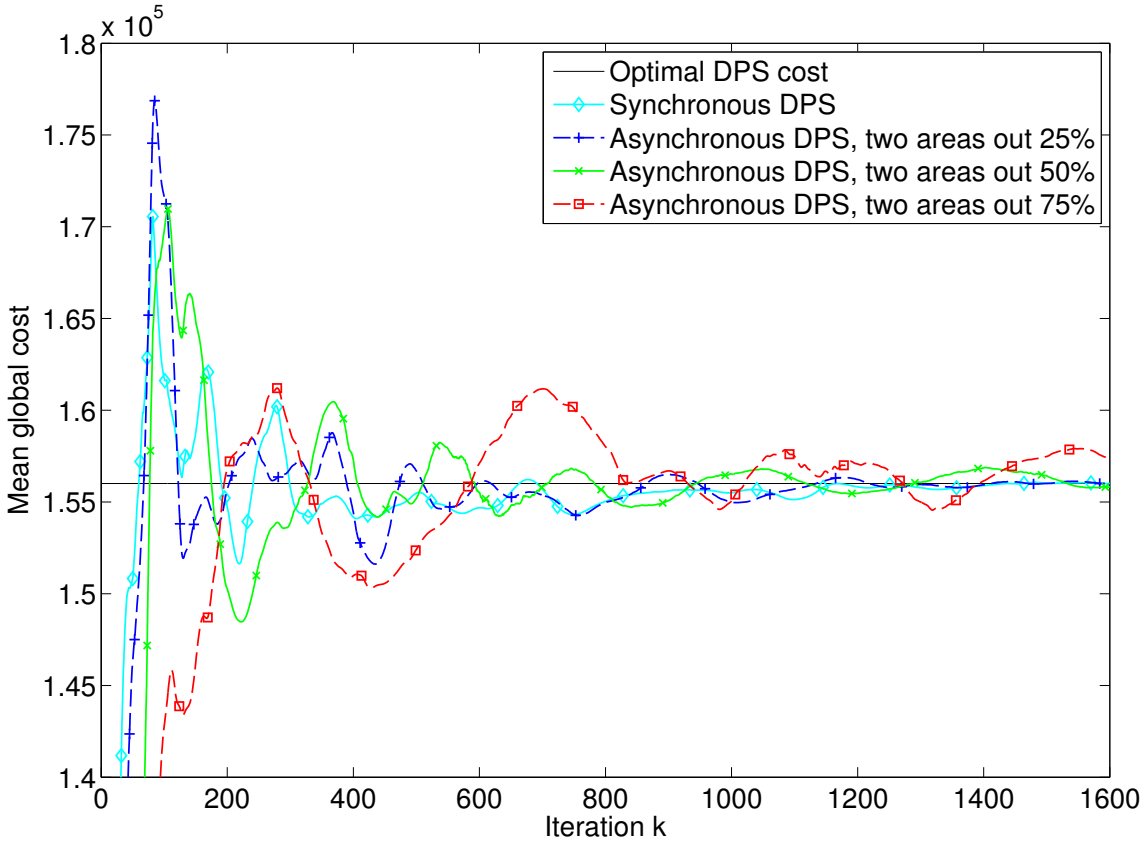

Figure 8.22: Mean global cost, IEEE−118 Bus system.

4. Asynchronous DPS, A_2 is 75% *off*: 2×10^{-5}.

When switching between the synchronous ADMM and the asynchronous ADMM case where A_2 is *off* for 75% of the time, the NMSD in Fig. 8.21, goes from 3×10^{-4} to 4.7×10^{-4}.

These small values demonstrate the convergence of the algorithms and its scalability property. Even when the network tends to be larger, the convergence is always guaranteed and achievable within a feasible number of iterations.

Two areas out

We implement the first two scenarios described in Section 8.3.1. A_1 is always activated and A_2 and A_3 are being activated on random basis for a total percentage of 25 then 50 and 70 respectively. Fig. 8.22 depicts the objective

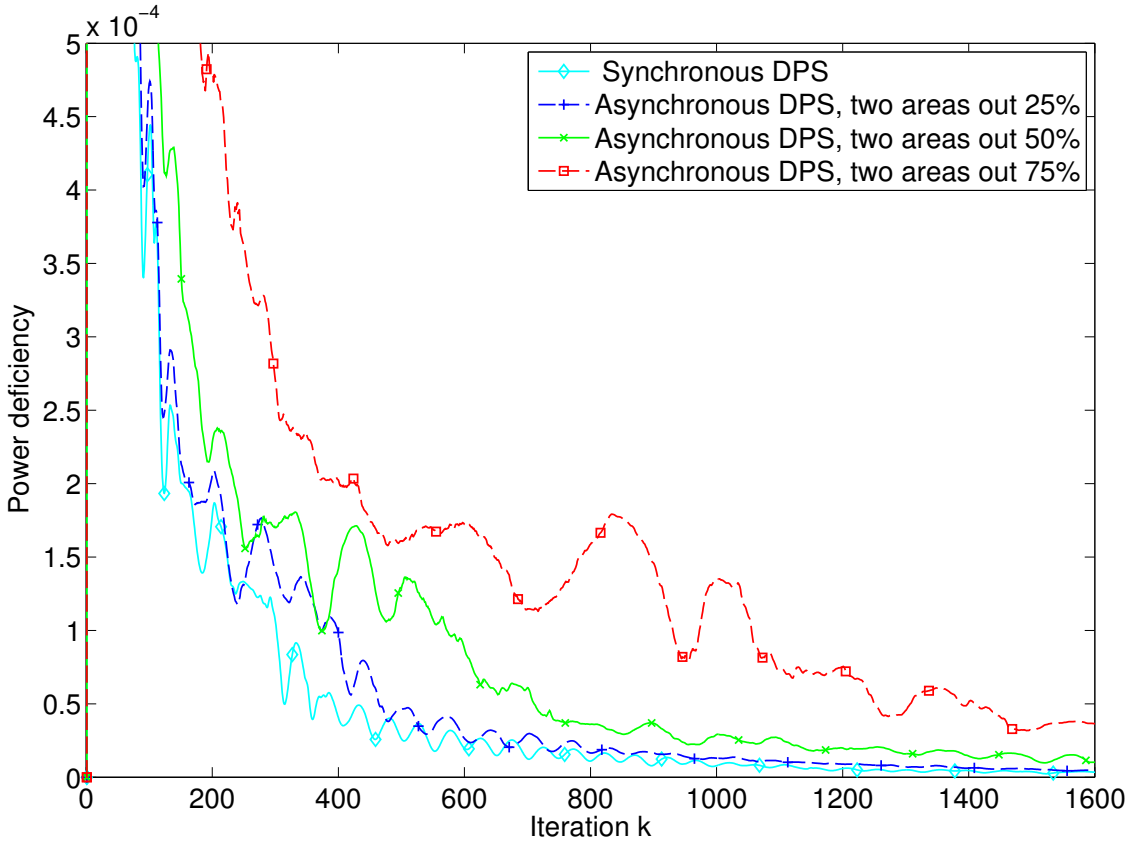

Figure 8.23: Mean global power deficiency, IEEE−118 Bus system.

convergence in each of these case compared to the synchronous distributed ADMM.

The relative error with respect to the optimal cost for each case after approximately 1000 iterations are as follows:

1. Synchronous DPS: 0.01%
2. Asynchronous DPS, A_2 and A_3 are 25% *off*: 0.06%
3. Asynchronous DPS,A_2 and A_3 are 50% *off*: 0.092%
4. Asynchronous DPS, A_2 and A_3 are 75% *off*: 0.2%.

The plots showing the residual convergence or mean global power deficiency and the normalized mean squared deviation NSMD are depicted by Fig. 8.23 and Fig. 8.24 respectively. The values of the NMSD and the mean

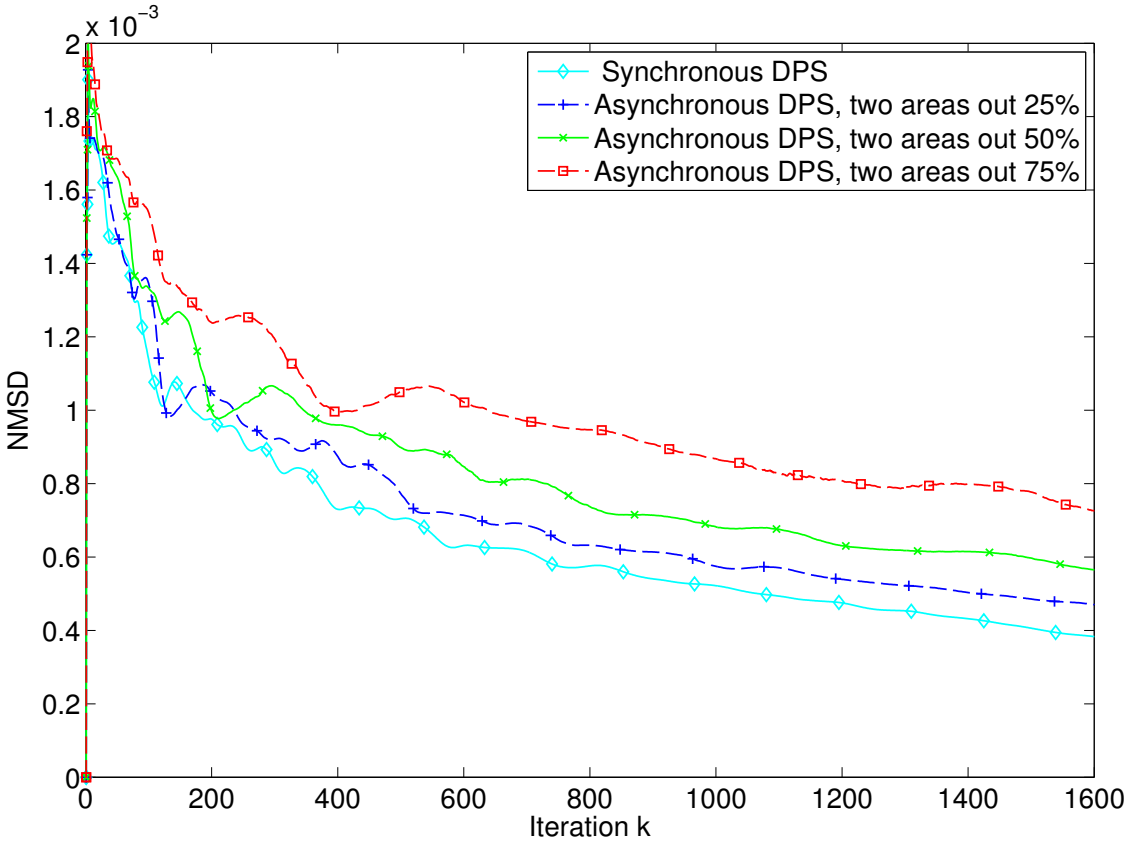

Figure 8.24: NMSD, IEEE−118 Bus system.

global power deficiency are well seen from each plot. From the synchronous ADMM to the asynchronous ADMM case where A_2 and A_3 are *off* for 75% of the time, the mean global power deficiency increase from 3×10^{-6} to 3×10^{-5} respectively. As for the NMSD, it scales from 3×10^{-4} to 7×10^{-4}.

These plots illustrate the capability of our algorithms to solve the DC-OPF problem even when the network gets larger. This feature makes them adequate for large power grid problems.

8.5 DC-OPF with renewable sources and storage devices

We now turn attention to the network with renewable sources and storage devices that was discussed in Chapter 7. We combine Lyapunov optimization

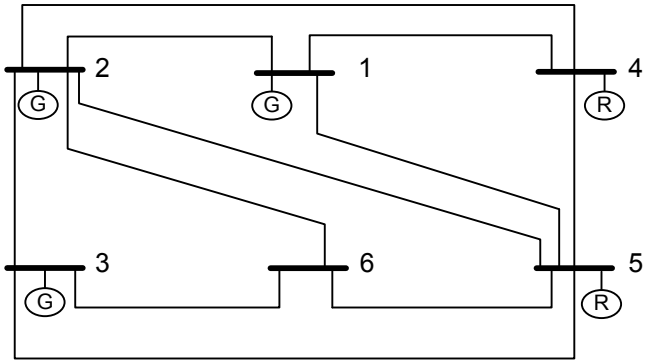

Figure 8.25: IEEE−6 bus test system with two storage devices.

and ADMM algorithm to fully distribute the production sharing problem while some of the node generate renewable energy and are equipped wit storage devices. We consider IEEE−6 Bus Test System given by Fig. 8.25. Then, we apply Algorithm 6 to this network. The IEEE−6 Bus Test System is composed of $N = 6$ buses and $m = 11$ branches [1].

We set time T to be equal to 10 time slots. The data related to the cost and limits on the conventional generated power are given in Table 8.6. We suppose that the network comprises 3 conventional generators, located at nodes $1, 2$ and 3.

Each node has a demand that varies over t, Table 8.7 gives the demand at each node for each time slot. The data related to the transmission lines are provided in [1].

We suppose that buses 4 and 5 are equipped with renewable energy sources and storage devices. The maximum generated power by the renewable sources placed at these nodes are given by Table 8.8.

We set the maximum storage capacity E_{max} to 100 p.u. for each storage device. We suppose that these devices have the same maximum charging rate $Y_{max,ch}$ and maximum discharging rate $Y_{max,dis}$. We check the effect of changing these rates.

The plots corresponding to the cost for each time slot t are depicted in Fig 8.26. The mean global cost over all the time slots with respect to the number of iterations is given in Fig 8.27. From these plots we can observe the convergence of the algorithm combining Lyapunov optimization with ADMM. Approximately 140 iterations were enough for the ADMM to converge. We can also observe the influence of increasing the maximum charging/discharging rates on the global cost. Indeed, when the renewable

Table 8.6: Conventional Generators data, IEEE−6 Bus Test System

Node j	$\underline{p_j^G}$	$\overline{p_j^G}$	c_j	c'_j
1	0	200	11,669	0,00533
2	0	150	10,333	0,00889
3	0	180	10,833	0,00741

Table 8.7: IEEE−6 Bus Test System Demand

Node	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$	$t=6$	$t=7$	$t=8$	$t=9$	$t=10$
1	50	110	100	80	120	90	104	86	80	80
2	80	100	100	90	70	80	0	100	90	0
3	90	100	0	90	120	0	100	0	80	0
4	50	130	160	80	136	80	104	86	80	0
5	50	130	100	80	106	80	64	86	80	0
6	70	0	50	80	0	90	0	80	80	100

Table 8.8: IEEE−6 Bus Test System Renewable Generation

Node	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$	$t=6$	$t=7$	$t=8$	$t=9$	$t=10$
4	90	110	130	180	180	90	70	60	50	80
5	86	86	65	120	117	127	80	80	90	90

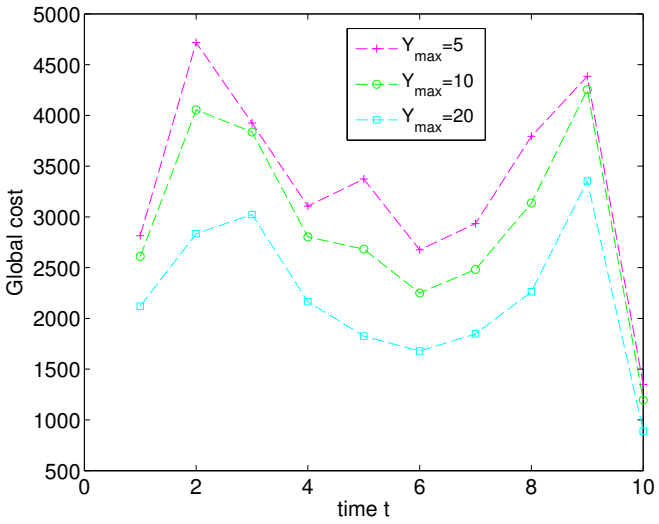

Figure 8.26: IEEE−6 bus test system with two storage devices, global cost.

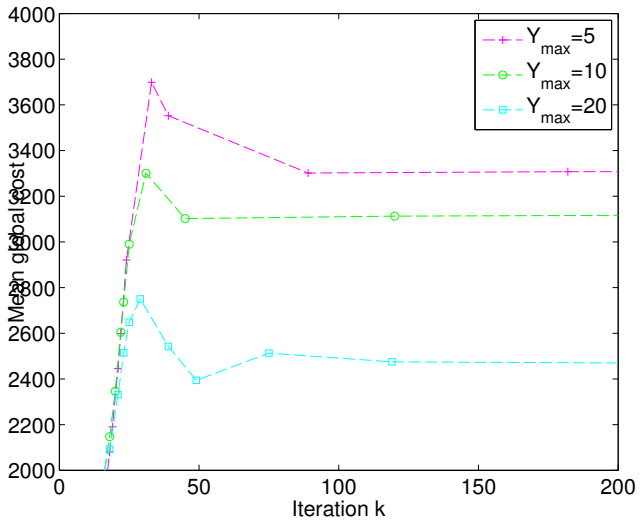

Figure 8.27: IEEE−6 bus test system with two storage devices objective convergence.

generation is greater than what is required and when $Y_{max,ch}$ and $Y_{max,dis}$ increase, the storage devices appear to have a bigger influence on decreasing the global cost over time.

8.6 Conclusion

We concluded the work accomplished in this part of the thesis with an implementation of the synchronous and asynchronous ADMM to the DC-OPF problem with and without storage devices and renewable energy sources. We tried to simulate multiple scenarios on different networks. We were able to track the convergence of the synchronous and asynchronous ADMM algorithms. Whether the network is small or large, whether the areas are distinct or overlapping, the algorithms converged to the optimal solution. The obtained solution meets all the constraints while providing the minimal global cost. Simulations were first carried out on the conventional power grid while considering the IEEE−30 bus test system [97] and the IEEE−118 bus test system. Then, for the implementation of ADMM on the DC-OPF with storage devices and renewable energy sources we used the IEEE−6 bus test system.

Chapter 9

Distributed Caching while Sharing in $5G$ Networks: An ADMM approach

9.1 Introduction

In this work we consider the problem of distributed caching in next generation mobile cellular networks (a.k.a., 5G) where densely-deployed small base stations (SBSs) are able to store and deliver users' content accordingly. In particular, we formulate the optimal cache allocation policy as a convex optimization problem where a subset of SBSs have their own i) local cost function which captures backhaul consumption aspects in terms of bandwidth and ii) a set of local network parameters and storage constraints iii) local cost on sharing contents with other SBSs. In this context, an SBS may privilege the option of sharing contents among its neighbors when the cost of sharing these contents is less than the cost of fetching the file from the Central Scheduler (CS). Given the fact that a coordination is involved between SBSs and due to the ability of ADMM to solve such a complicated optimization problem in a simplified manner, we provide a distributed solution for the caching problem using ADMM. The ADMM application in such context converges and the optimal solution for the caching problem is achieved.

9.2 Related work

Ever growing demand of mobile users [19] is reshaping discussions both in industry and academia, pushing the current mobile infrastructure to evolve towards next generation (a.k.a. $5G$) mobile cellular networks [20]. One of

the candidate solutions to satisfy this demand and offload the backhaul is to proactively store users' contents at the edge of the mobile network, either in base stations or user terminals [98].

Indeed, although the idea of putting users' contents in cache-enabled nodes of the cellular networks is somewhat recent, many works have addressed the caching problem from different aspects resulting in an extensive literature. For instance, predicting users' behavior and proactively storing their contents at SBS is studied in [99], whereas the benefit of proactive caching in a mobility setup is exploited in [100]. A coded caching scheme using information theoretic arguments is given in [101], whereas a similar scheme but with multi-level architecture is studied in [102]. Performance evaluation of coded caching gains setups can be found in [103–105]. From a game theoretic standpoint, a many-to-many matching game formulation which takes into consideration the content dissemination in social networks is shown in [106]. Under a given estimation of backhaul usage via CF, a one-to-many matching game between SBSs and UT is formulated in [107]. Additionally, a learning based online caching scheme is presented in [108]. These studies point out the importance of caching in $5G$ wireless networks, and provide their investigations both from performance and algorithmic perspectives. However, the practical efforts for these dense networks are still in its infancy mainly due to lack of easy-to-implement distributed solutions.

Given the motivations above, our main contribution in this work is to formulate the cache allocation policy as a convex optimization problem and provide a distributed algorithm implemented at each cache-enabled SBS. More specifically, we define a global convex cost function as the sum of local convex functions of users' demand and network topology (i.e., physical connection between SBSs and UTs) and the linear constrains are given on sharing a file and its availability; and storage size due to the resource-limited SBSs. To solve this problem, we adopt a similar approach to 5 where we used ADMM in the context of optimal power flow in smart grids. This allows each SBS to solve its given sub-problem via this low-complexity iterative algorithm by taking into account users' demand, network topology and storage constraint.

Briefly, the motivation of using such a distributed approach is to 1) avoid the communication overhead between SBSs and the CS that is in charge of the decision mechanism, and 2) distribute the computational burden of the CS among SBSs. Indeed, the origin of distributed optimization techniques dates back to the seminal work of Tsitsiklis and Bertsekas [109]. Among extensive studies on these techniques which are not covered in this work due to the lack of space, ADMM is shown to promise faster convergence at some negligible cost of synchronization and coordination compared to its alternatives [70].

9.3 Network Model

Let us consider a network consisting of a set of M SBSs and N users denoted by $\mathcal{M} = \{1, ..., M\}$ and $\mathcal{N} = \{1, ..., N\}$ respectively. In this setup, the SBSs are connected to a CS via limited backhaul links with the purpose of providing broadband Internet connection to their users. The wireless downlink rates from the SBSs to users are given by the matrix $\mathbf{R}$ of dimension $M \times N$, where each entry $R_{m,n}$ denotes the achievable rate from SBS m to user n. For simplicity, we assume that communication in the downlink is done in a TDD manner with multiple frequency blocks (i.e., OFDMA), thus intra-cell and inter-cell interference are avoided. Then, the *wireless connectivity matrix* $\mathbf{O} \in \{0, 1\}^{M \times N}$, representing the connections between the SBSs and users is structured as

$$\mathbf{O} = \begin{bmatrix} \mathbf{o}_1 \\ \vdots \\ \mathbf{o}_M \end{bmatrix} = \begin{bmatrix} o_{1,1} & \dots & o_{1,N} \\ \vdots & \ddots & \vdots \\ o_{M,1} & \dots & o_{M,N} \end{bmatrix}, \tag{9.1}$$

where $o_{m,n} = \mathbf{1}\{R_{m,n} \geq R'\}$. The purpose of the target bit-rate R' constraint for connectivity is to guarantee a certain Quality of Service (QoS) in the downlink. Each user n with a wireless link rate $R_{m,n}$ below this threshold is assumed to be not connected to the SBS m.

In our model, we assume that the SBSs have storage units with capacities $\mathbf{b} = [b_1, ..., b_M] \in \{\mathbb{Z}^+\}^{1 \times M}$. These storage capacities in the decreasing ordered case follow a Zipf-like distribution $P_{\mathbf{b}}(s, \alpha)$ defined as [110]

$$P_{\mathbf{b}}(b, \alpha) = \frac{\Omega}{b^\alpha} \tag{9.2}$$

with

$$\Omega = \Big(\sum_{i=1}^{M} \frac{1}{i^\alpha} \Big)^{-1},$$

where the parameter α characterizes the steepness of the distribution. In fact, the evidence of such a law is shown for the content distribution in web proxies [110], whereas in our case we also treat for modeling the storage capacity distribution. By having storage capabilities at the SBSs, contents can be cached in order to serve users' predicted requests locally, and thus reduce backhaul usage and access delays. The sketch of the considered network model is given in Fig. 9.1.

In the following, we suppose that the users' content demand arrival, over T time slots, is modeled by a Poisson process with rate/intensity parameter λ. Additionally, we suppose that users' demands are made from a *catalog* of F

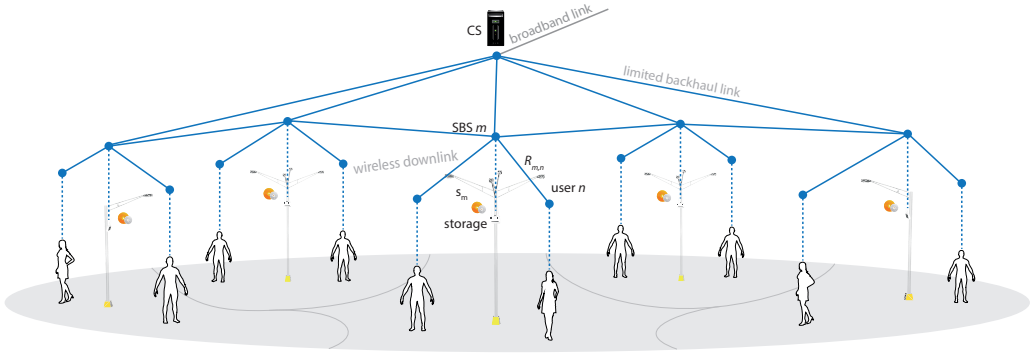

Figure 9.1: An illustration of the scenario which consists of M cache-enabled SBSs and N users.

distinct contents. The length of a file f is denoted by $s_f < b_m, m = 1, ..., M$. We assume that the content popularity distribution of users' drawn from the catalog is characterized by another Zipf law with parameter β. Given the arrival process and content popularity distribution, the users' content demand counts are represented by the *user demand matrix* $\mathbf{D}^u \in \{\mathbb{N}\}^{N \times F}$. Then, the whole content demands observed at the SBSs level is given by

$$\mathbf{D} = \mathbf{O}\mathbf{D}^u \in \{0, 1\}^{M \times F}. \tag{9.3}$$

For ease of exposition, we assume that the matrix $\mathbf{D}$ is perfectly known. Note that, in practice, the demand of users (and its observation at SBSs) are correlated and can be predicted up to a certain level, i.e., using statistical inference tools from machine learning [98]. We use d_{jf} to point at the demand of a file f at the SBS j.

The cache indicator vector at node $j \in 1, \ldots, M$ is given as follows

$$\begin{aligned} \boldsymbol{x_j} = (&x_{j01}, x_{j02}, \quad \ldots, x_{j0F}, \quad x_{j11}, x_{j12}, \quad \ldots, \\ &x_{j1F}, x_{j12}, \quad \ldots, x_{jM1}, \quad x_{jM2}, \ldots, \quad x_{jMF})^1 \end{aligned} \tag{9.4}$$

where, x_{j0f} is the variable related to fetching the content from the CS for every user's request, the value of x_{jjf} indicates whether the f^{th} content is going to be cached at SBS j or not, and x_{jif} points to the fact that node j will obtain the f^{th} content from node i.

For this sharing problem to be consistent, we need to ensure the following:

1. The content sharing from i to j may exist only if node i is going to cache the content

$$x_{jin} \leq x_{iin}, \forall j, i > 0 \,\&\, i \neq j \tag{9.5}$$

2. The storage capacity can never be exceeded for each node

$$\sum_{f=1}^{F} s_f \, x_{jjn} \leq b_j, \, \forall j \tag{9.6}$$

3. A user request has to be met, either by fetching, caching or sharing the file

$$x_{jjf} + \sum_{i=0, i \neq j}^{M} x_{jif} \leq 1, \, \forall j \tag{9.7}$$

The equality is reached when the algorithm converges and users are satisfied.

9.4 Problem formulation

We consider that all the nodes have the ability to share files. The value assigned to the cost of file sharing between two nodes plays an important role on leveraging certain connections on others. Additionally, when considering a very high cost on a certain link, it is equivalent to canceling that link due to its high cost. In this way we can tune our model to special network structures.

The costs of sharing, fetching or caching a file are given as follows:

1. The cost of caching a content f,

$$s_f \, c_{j0} \, x_{jjf} \tag{9.8}$$

2. The cost of fetching a file f every time i.e., not caching the file,

$$c_{j0} \, d_{jf} \, s_f \, x_{j0f} \tag{9.9}$$

3. The cost of sharing a file f from another node i ,

$$c_{ji} \, s_f \, d_{jf} \, x_{jif} \tag{9.10}$$

Thus, the total cost for each node j is given by:

$$\sum_{f=1}^{F} \left\{ c_{j0} \, d_{jf} \, s_f \, x_{j0f} + c_{j0} \, s_f \, x_{jjf} + \sum_{i=1, i \neq j}^{M} \left(c_{ji} \, s_f \, d_{jf} \, x_{ijf} \right) \right\} \tag{9.11}$$

Let $h_j(\boldsymbol{x}_j)$ represent the local cost at node j, we can write it down as:

$$h_j(\boldsymbol{x}_j) = \sum_{f=1}^{F} \left\{ c_{i0}\, s_f\, x_{iif} + \sum_{i=0, j\neq j}^{M} (c_{ji}\, s_f\, d_{jf}\, x_{jif}) \right\} \tag{9.12}$$

The caching while sharing problem in 5G network can be written formally as:

$$\begin{aligned} \min_{\substack{\boldsymbol{x}_j \in \mathbb{R}^{F(M+1)} \\ j=1,\ldots,M}} \quad & \sum_{j=1}^{M} h_j(\boldsymbol{x}_j) \\ \text{subject to} \quad & \begin{cases} x_{jif} \leq x_{iif},\ \forall j > 0, i > 0, i \neq j \\ \sum_{f=1}^{F} s_f x_{jjf} \leq b_j,\ \forall j > 0 \\ x_{jjf} + \sum_{i=0, i\neq j}^{M} x_{jif} \leq 1,\ \forall j > 0, \forall f = 1, \ldots, F \end{cases} \end{aligned} \tag{9.13}$$

We relax the values of x_{ijf} and we use three sets of slack variables $\boldsymbol{sl}_1 \in \mathcal{R}_+^{FM(M-1)}; \boldsymbol{sl}_2 \in \mathcal{R}_+^{M}; \boldsymbol{sl}_3 \in \mathcal{R}_+^{FM}$ in order to transform the inequalities into equality constraints as follows:

$$\begin{aligned} \min_{\substack{\boldsymbol{x}_j \in \mathbb{R}^{F(M+1)} \\ j=1,\ldots,M}} \quad & \sum_{j=1}^{M} h_j(\boldsymbol{x}_j) \\ \text{subject to} \quad & \begin{bmatrix} \mathbf{A_1} & \mathbf{A_2} & \mathbf{A_3} & \mathbf{A_4} \\ \mathbf{A_5} & \mathbf{A_6} & \mathbf{A_7} & \mathbf{A_8} \\ \mathbf{A_9} & \mathbf{A_{10}} & \mathbf{A_{11}} & \mathbf{A_{12}} \end{bmatrix} \boldsymbol{u} = \tilde{\boldsymbol{b}} \end{aligned} \tag{9.14}$$

where

$$\boldsymbol{u} = [\boldsymbol{x}_1; \boldsymbol{x}_2; \ldots; \boldsymbol{x}_M; \boldsymbol{sl}_1; \boldsymbol{sl}_2; \boldsymbol{sl}_3]$$

and

$$\tilde{\boldsymbol{b}} = [0; 0; \ldots; 0; b_1; \ldots; b_M; 1; \ldots; 1].$$

$\mathbf{A_1}$ is an $MF(M-1) \times MF(M+1)$ matrix, $\mathbf{A_2}$ is an $MF(M-1) \times MF(M-1)$ matrix, $\mathbf{A_3}$ is an $MF(M-1) \times M$ matrix, $\mathbf{A_4}$ is an $MF(M-1) \times MF$ matrix.

$\mathbf{A_5}$ is an $M \times MF(M+1)$ matrix, $\mathbf{A_6}$ is an $M \times MF(M-1)$ matrix, $\mathbf{A_7}$ is an $M \times M$ matrix, $\mathbf{A_8}$ is an $M \times MF$ matrix.

$\mathbf{A_9}$ is an $MF \times MF(M+1)$ matrix, $\mathbf{A_{10}}$ is an $MF \times MF(M-1)$ matrix, $\mathbf{A_{11}}$ is an $MF \times M$ matrix, $\mathbf{A_{12}}$ is an $MF \times MF$ matrix.

A simple example: For illustration purposes we consider the case of $M = 3$ nodes and $F = 3$ files. The results can be easily generalized to any

other number of nodes and contents.

$$\mathbf{A_1} = \begin{bmatrix} 0_F & 0_F & I_F & 0_F & 0_F & 0_F & -I_F & 0_F & 0_F & 0_F & 0_F & 0_F \\ 0_F & 0_F & 0_F & I_F & 0_F & 0_F & 0_F & 0_F & 0_F & 0_F & 0_F & -I_F \\ 0_F & -I_F & 0_F & 0_F & 0_F & I_F & 0_F & 0_F & 0_F & 0_F & 0_F & 0_F \\ 0_F & 0_F & 0_F & 0_F & 0_F & 0_F & 0_F & I_F & 0_F & 0_F & 0_F & -I_F \\ 0_F & -I_F & 0_F & 0_F & 0_F & 0_F & 0_F & 0_F & 0_F & I_F & 0_F & 0_F \\ 0_F & 0_F & 0_F & 0_F & 0_F & 0_F & -I_F & 0_F & 0_F & 0_F & I_F & 0_F \end{bmatrix}$$

$$\mathbf{A_2} = \begin{bmatrix} I_F & 0_F & 0_F & 0_F & 0_F & 0_F \\ 0_F & I_F & 0_F & 0_F & 0_F & 0_F \\ 0_F & 0_F & I_F & 0_F & 0_F & 0_F \\ 0_F & 0_F & 0_F & I_F & 0_F & 0_F \\ 0_F & 0_F & 0_F & 0_F & I_F & 0_F \\ 0_F & 0_F & 0_F & 0_F & 0_F & I_F \end{bmatrix}$$

$$\mathbf{A_3} = \begin{bmatrix} 0_{MF(M-1)\times M} \end{bmatrix}$$

$$\mathbf{A_4} = \begin{bmatrix} 0_{MF(M-1)\times MF} \end{bmatrix}$$

$$\mathbf{A_5} = \begin{bmatrix} 0_{1\times F} & 1_{1\times F} & 0_{1\times F} & 0_{1\times F} & 0_{1\times F} & 0_{1\times F} & 0_{1\times F} & 0_{1\times F} & 0_{1\times F} & 0_{1\times F} & 0_{1\times F} & 0_{1\times F} \\ 0_{1\times F} & 0_{1\times F} & 0_{1\times F} & 0_{1\times F} & 0_{1\times F} & 0_{1\times F} & 1_{1\times F} & 0_{1\times F} & 0_{1\times F} & 0_{1\times F} & 0_{1\times F} & 0_{1\times F} \\ 0_{1\times F} & 0_{1\times F} & 0_{1\times F} & 0_{1\times F} & 0_{1\times F} & 0_{1\times F} & 0_{1\times F} & 0_{1\times F} & 0_{1\times F} & 0_{1\times F} & 0_{1\times F} & 1_{1\times F} \end{bmatrix} o\, \boldsymbol{s_F}$$

$$\mathbf{A_6} = \begin{bmatrix} 0_{M\times MF(M-1)} \end{bmatrix}$$

$$\mathbf{A_7} = \begin{bmatrix} I_M \end{bmatrix}$$

$$\mathbf{A_8} = \begin{bmatrix} 0_{M\times MF} \end{bmatrix}$$

$$\mathbf{A_9} = \begin{bmatrix} I_F & I_F & I_F & I_F & 0_F & 0_F & 0_F & 0_F & 0_F & 0_F & 0_F & 0_F \\ 0_F & 0_F & 0_F & 0_F & I_F & I_F & I_F & I_F & 0_F & 0_F & 0_F & 0_F \\ 0_F & 0_F & 0_F & 0_F & 0_F & 0_F & 0_F & 0_F & I_F & I_F & I_F & I_F \end{bmatrix}$$

$$\mathbf{A_{10}} = \begin{bmatrix} 0_{MF\times MF(M-1)} \end{bmatrix}$$

$$\mathbf{A_{11}} = \begin{bmatrix} 0_{MF\times M} \end{bmatrix}$$

$$\mathbf{A_{12}} = \left[\; I_{MF} \; \right].$$

The nonzero elements in matrix $\mathbf{A_5}$ are equal to the size s_f of the corresponding content. $\boldsymbol{s_F}$ is the vector containing the size of all the files. It can be observe from these matrices that the problem scales with the number of nodes M ans the catalog size F.

The final global formulation for this problem:

$$\begin{aligned} \min_{\mathbf{u}} \quad & F(\boldsymbol{u}) = \sum_{j=1}^{M} h_j(\boldsymbol{x}_j) + \mathcal{I}_+(\boldsymbol{sl}_1) + \mathcal{I}_+(\boldsymbol{sl}_2) + \mathcal{I}_+(\boldsymbol{sl}_3) \\ \text{subject to} \quad & \mathbf{A}\boldsymbol{u} = \tilde{\boldsymbol{b}} \end{aligned} \tag{9.15}$$

$\mathbf{A}$ is an $M(MF+1) \times M(2FM+F+1)$ matrix and $\boldsymbol{u}$ is the $M(2FM+F+1)$ vector comprising the primal and slack variables.

Conventionally, this problem can be solved at the CS in a centralized way by

i) collecting users' demands from the SBSs as well as the storage capacities of SBSs;

ii) solving the problem accordingly;

iii) transferring the adequate cache indicator vector to each SBS.

This centralized optimization scheme however induces latency in the computation and communication burden. Alternatively, the solution of such a problem can be obtained efficiently by using distributed convex optimization solvers. In the next section, we detail the steps of the ADMM-based approach, which in turn will allow us to handle the problem at each SBSs distributively.

9.5 Optimal Cache w/ ADMM

Problem (9.15) is similar to the Production-Sharing problem described in Chapter 4. In the same manner we reformulate the problem as a distributed problem to which we can apply ADMM. We follow the steps used to obtain Algorithm 1.

Let $\boldsymbol{v}$ be the vector of auxiliary variables introduced by the algorithm. $\boldsymbol{v}$ have the same size $M' \hat{=} M^2(MF+1)(2FM+F+1)$ as the Lagrangian multipliers vector $\boldsymbol{\lambda}$.

Let C be the indicator function C on $\boldsymbol{v}$ defined as:

$$\begin{array}{rccl} C: & \mathbb{R}^{M')} & \longrightarrow & (-\infty, +\infty] \\ & \boldsymbol{v} & \longmapsto & \begin{cases} 0 \quad \text{if } \forall i = 1 \dots M(MF+1), \sum\limits_{(i-1)M(2MF+F+1)+1}^{iM(2MF+F+1)} v_j = \tilde{b}_i \\ +\infty \quad \text{elsewhere} \end{cases} \end{array} \tag{9.16}$$

where

$$\tilde{\mathbf{A}} = \begin{bmatrix} \text{diag}([\mathbf{A}]_1) \\ \vdots \\ \text{diag}([\mathbf{A}]_{M(MF+1)}) \end{bmatrix} \tag{9.17}$$

Finally, we obtain a problem similar to the DPS problem (4.16) that we rewrote as:

$$\min_{\boldsymbol{u}} \quad F(\boldsymbol{u}) + C(\tilde{\mathbf{A}}\boldsymbol{u}) \tag{9.18}$$

To this problem we apply the ADMM in order to obtain a distributed algorithm solved at node level. The iterations resulting from the application of ADMM to problem (9.15) are as follows:

$$\boldsymbol{u}^{k+1} = \underset{\boldsymbol{y}}{\text{argmin}}\ \mathcal{L}_\rho(\boldsymbol{u}, \boldsymbol{v}^k; \boldsymbol{\lambda}^k), \tag{9.19a}$$

$$\boldsymbol{v}^{k+1} = \underset{\boldsymbol{v}}{\text{argmin}}\ \mathcal{L}_\rho(\boldsymbol{u}^{k+1}, \boldsymbol{v}; \boldsymbol{\lambda}^k), \tag{9.19b}$$

$$\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + \rho(\boldsymbol{A}\boldsymbol{u}^{k+1} - \boldsymbol{v}^{k+1}). \tag{9.19c}$$

In the following, we reference the elements of $\boldsymbol{u}$ either as u_j or using the original variable that they represent.

The primal variables iteration (9.19a) can be expanded using the corresponding entries of $\boldsymbol{v}$ and $\boldsymbol{\lambda}$ indexed by where $\beta \hat{=} \beta(i,j) = M(i-1)(2MF + F + 1) + j$ as follows:

$$\begin{aligned}\boldsymbol{u}^{k+1} = &\underset{\boldsymbol{u}}{\operatorname{argmin}}\ F(\boldsymbol{u})+ \\ &\sum_{i=1}^{M(MF+1)} \frac{\rho}{2} \sum_{j=1}^{MF(M+1)} \left\| A_{ij}u_j + \frac{\lambda_\beta^k}{\rho} - v_\beta^k \right\|^2 + \\ &\sum_{i=1}^{MF(M-1)} \frac{\rho}{2} \sum_{j=1}^{MF(M-1)} \left\| A_{ij_{+MF(M+1)}} sl_{1,i} + \frac{\lambda_{\beta+MF(M+1)}^k}{\rho} - v_{\beta+MF(M+1)}^k \right\|^2 + \\ &\sum_{i=MF(M-1)+1}^{MF(M-1)+M} \frac{\rho}{2} \sum_{j=1}^{M} \left\| A_{ij_{+MF(2M)}} sl_{2,j} + \frac{\lambda_{\beta+MF(2M)}^k}{\rho} - v_{\beta+MF(2M)}^k \right\|^2 + \\ &\sum_{i=MF(M-1)+M+1}^{M(MF+1)} \frac{\rho}{2} \sum_{j=1}^{MF} \left\| A_{ij_{+MF(2M)+M}} sl_{3,j} + \frac{\lambda_{\beta+MF(2M)+M}^k}{\rho} - v_{\beta+MF(2M)+M}^k \right\|^2 .\end{aligned} \tag{9.20}$$

As for the auxiliary variable iteration (9.19b), we can reformulate it as follows:

$$\begin{aligned}\boldsymbol{v}^{k+1} = &\underset{\boldsymbol{v}}{\operatorname{argmin}}\ C(\boldsymbol{v})+ \\ &\sum_{i=1}^{M(MF+1)} \frac{\rho}{2} \sum_{j=1}^{MF(M+1)} \left\| A_{ij}u_j^{k+1} + \frac{\lambda_\beta^k}{\rho} - v_\beta \right\|^2 + \\ &\sum_{i=1}^{MF(M-1)} \frac{\rho}{2} \sum_{j=1}^{MF(M-1)} \left\| A_{ij_{+MF(M+1)}} sl_{1,i}^{k+1} + \frac{\lambda_{\beta+MF(M+1)}^k}{\rho} - v_{\beta+MF(M+1)} \right\|^2 + \\ &\sum_{i=MF(M-1)+1}^{MF(M-1)+M} \frac{\rho}{2} \sum_{j=1}^{M} \left\| A_{ij_{+MF(2M)}} sl_{2,j}^{k+1} + \frac{\lambda_{\beta+MF(2M)}^k}{\rho} - v_{\beta+MF(2M)} \right\|^2 + \\ &\sum_{i=MF(M-1)+M+1}^{M(MF+1)} \frac{\rho}{2} \sum_{j=1}^{MF} \left\| A_{ij_{+MF(2M)+M}} sl_{3,j}^{k+1} + \frac{\lambda_{\beta+MF(2M)+M}^k}{\rho} - v_{\beta+MF(2M)+M} \right\|^2 \\ &\text{subject to} \sum_{j=(i-1)M(2MF+F+1)+1}^{iM(2MF+F+1)} v_j = \tilde{b}_i, \forall i = 1, \ldots, M(MF+1).\end{aligned} \tag{9.21}$$

We start by decomposing this vector (9.21) into a series of 4 components

corresponding to $\boldsymbol{x}_i, \boldsymbol{sl}_1, \boldsymbol{sl}_2$, and $\boldsymbol{sl}_3$ respectively.

$$v_\beta^{k+1} = \underset{v_\beta}{\operatorname{argmin}} \left\| A_{ij} u_j^{k+1} + \frac{\lambda_\beta^k}{\rho} - v_\beta \right\|^2 \tag{9.22a}$$

$$v_{\beta_{+MF(M+1)}}^{k+1} = \underset{v_{\beta_{+MF(M+1)}}^{k+1}}{\operatorname{argmin}} \left\| A_{ij_{+MF(M+1)}} sl_{1,i}^{k+1} + \frac{\lambda_{\beta+MF(M+1)}^k}{\rho} - v_{\beta+MF(M+1)} \right\|^2 \tag{9.22b}$$

$$v_{\beta_{+MF(2M)}}^{k+1} = \underset{v_{\beta_{+MF(2M)}}^{k+1}}{\operatorname{argmin}} \left\| A_{ij_{+MF(2M)}} sl_{2,j}^{k+1} + \frac{\lambda_{\beta+MF(2M)}^k}{\rho} - v_{\beta+MF(2M)} \right\|^2 \tag{9.22c}$$

$$v_{\beta_{+M(2MF+1)}}^{k+1} = \underset{v_{\beta_{+M(2MF+1)}}^{k+1}}{\operatorname{argmin}} \left\| A_{ij_{+MF(2M)+M}} sl_{3,j}^{k+1} + \frac{\lambda_{\beta+MF(2M)+M}^k}{\rho} - v_{\beta+MF(2M)+M} \right\|^2 \tag{9.22d}$$

$$\text{subject to} \sum_{j=(i-1)M(2MF+F+1)+1}^{iM(2MF+F+1)} v_j = \tilde{b}_i, \forall i = 1, \ldots, M(MF+1).$$

We introduce the $M(MF+1)$ Lagrangian multipliers vector $\boldsymbol{\pi}$ on the condition linking the entries of $\boldsymbol{v}$, this helps in further simplifying its components as follows for each $i = 1, \ldots, M(MF+1)$:

$$v_\beta^{k+1} = A_{ij} u_j^{k+1} + \frac{\lambda_\beta^k - \pi_i^{k+1}}{\rho} \tag{9.23a}$$

$$v_{\beta_{+MF(M+1)}}^{k+1} = A_{ij_{+MF(M+1)}} sl_{1,i}^{k+1} + \frac{\lambda_{\beta+MF(M+1)-\pi_i^{k+1}}^k}{\rho} \tag{9.23b}$$

$$v_{\beta_{+MF(2M)}}^{k+1} = A_{ij_{+MF(2M)}} sl_{2,j}^{k+1} + \frac{\lambda_{\beta+MF(2M)}^k - \pi_i^{k+1}}{\rho} \tag{9.23c}$$

$$v_{\beta_{+M(2MF+1)}}^{k+1} = A_{ij_{+M(2MF+1)}} sl_{3,j}^{k+1} + \frac{\lambda_{\beta_{+M(2MF+1)}}^k - \pi_i^{k+1}}{\rho}. \tag{9.23d}$$

We can decompose $\boldsymbol{\lambda}$ into 4 sets of components corresponding to those

indexed for $\boldsymbol{v}$. Using (9.23a) we can write:

$$\begin{aligned}\lambda_\beta^{k+1} &= \lambda_\beta^k + \rho(A_{ij}u_j^{k+1} - v_\beta^{k+1}) \\ &= \lambda_\beta^k + \rho(A_{ij}u_j^{k+1} - A_{ij}u_j^{k+1} - \frac{\lambda_\beta^k - \pi_i^{k+1}}{\rho} \\ &= \pi_i^{k+1}. \end{aligned} \tag{9.24}$$

The same reasoning can be used for the other components of $\boldsymbol{\lambda}$. Thus, for each $i = 1, \ldots, M(MF+1)$ we have:

$$\lambda_\beta^{k+1} = \lambda_{\beta_{+MF(M+1)}}^{k+1} = \lambda_{\beta_{+MF(2M)}}^{k+1} = \lambda_{\beta_{+M(2MF+1)}}^{k+1} = \pi_i^{k+1}. \tag{9.25}$$

This result can be helpful in obtaining the update step of π_i, we start with the constraint on the auxiliary variables:

$$\begin{aligned}\sum_{j=(i-1)M(2MF+F+1)+1}^{iM(2MF+F+1)} v_j^{k+1} &= \tilde{b}_i \\ \sum_{j=(i-1)M(2MF+F+1)+1}^{iM(2MF+F+1)} \{A_{ij}u_j^{k+1} + \frac{\pi_i^k - \pi_i^{k+1}}{\rho}\} &= \tilde{b}_i. \end{aligned} \tag{9.26}$$

We obtain the Lagrangian variable update step:

$$\pi_i^{k+1} = \pi_i^k + \frac{\rho}{d(i)} r_i(\boldsymbol{u}^{k+1}) \tag{9.27}$$

where,

$$r_i(\boldsymbol{u}^{k+1}) = \sum_{j=(i-1)M(2MF+F+1)+1}^{iM(2MF+F+1)} \{A_{ij}u_j^{k+1}\} - \tilde{b}_i \tag{9.28}$$

is the residual of the i^{th} constraint.

Given these results, we can rewrite the auxiliary variables updates steps as:

$$v_\beta^{k+1} = A_{ij}u_j^{k+1} - \frac{r_i(\boldsymbol{u}^{k+1})}{d(i)} \tag{9.29a}$$

$$v_{\beta_{+MF(M+1)}}^{k+1} = A_{ij_{+MF(M+1)}} sl_{1,i}^{k+1} - \frac{r_i(\boldsymbol{u}^{k+1})}{d(i)} \tag{9.29b}$$

$$v_{\beta_{+MF(2M)}}^{k+1} = A_{ij_{+MF(2M)}} sl_{2,j}^{k+1} - \frac{r_i(\boldsymbol{u}^{k+1})}{d(i)} \tag{9.29c}$$

$$v_{\beta_{+M(2MF+1)}}^{k+1} = A_{ij_{+M(2MF+1)}} sl_{3,j}^{k+1} - \frac{r_i(\boldsymbol{u}^{k+1})}{d(i)}. \tag{9.29d}$$

At this point, we have managed to reduce vectors $\boldsymbol{\lambda}$ and $\boldsymbol{v}$. We return our attention to $\boldsymbol{u}$, this iteration (9.20) can be divided to 5 sets of updates.

For each node $j = 1, \ldots, M$

$$x_{j0f}^{k+1} = \underset{x_{j0f}}{\operatorname{argmin}} \frac{\rho}{2} \left\| x_{j0f} + \frac{\lambda_\beta^k}{\rho} - v_\beta^k \right\|^2 + c_{j0} s_f d_{jf} x_{j0f} \tag{9.30a}$$

$$x_{jjf}^{k+1} = \underset{x_{jjf}}{\operatorname{argmin}} \frac{\rho}{2} \left\| s_f x_{jjf} + \frac{\lambda_\beta^k}{\rho} - v_\beta^k \right\|^2 + \frac{\rho}{2} \left\| x_{jjf} + \frac{\lambda_\beta^k}{\rho} - v_\beta^k \right\|^2 + c_{j0} s_f x_{jjf} \tag{9.30b}$$

$$x_{jj'f}^{k+1} = \underset{x_{jj'f}}{\operatorname{argmin}} \frac{\rho}{2} \left\| x_{jj'f} + \frac{\lambda_\beta^k}{\rho} - v_\beta^k \right\|^2 + \frac{\rho}{2} \left\| x_{jj'f} + \frac{\lambda_\beta^k}{\rho} - v_\beta^k \right\|^2 + c_{jj'} s_f d_{jf} x_{jj'f} \tag{9.30c}$$

$$sl_{2,j}^{k+1} = \underset{sl_{2,j}}{\operatorname{argmin}} \mathcal{I}_+(sl_{2,j}) + \frac{\rho}{2} \left\| sl_{2,j} + \frac{\lambda_{\beta_{2+MF(2M)}}^k}{\rho} - v_{\beta_{2+MF(2M)}}^k \right\|^2. \tag{9.30d}$$

For each constraint $i = 1, \ldots, MF(M-1)$

$$sl_{1,i}^{k+1} = \underset{sl_{1,i}}{\operatorname{argmin}} \mathcal{I}_+(sl_{1,i}) + \frac{\rho}{2} \left\| sl_{1,i} + \frac{\lambda_{\beta_{1+MF(M+1)}}^k}{\rho} - v_{\beta_{1+MF(M+1)}}^k \right\|^2. \tag{9.31a}$$

For each constraint $i = 1, \ldots, MF$[2]

$$sl_{3,i}^{k+1} = \underset{sl_{3,i}}{\operatorname{argmin}} \mathcal{I}_+(sl_{3,i}) + \frac{\rho}{2} \left\| sl_{3,i} + \frac{\lambda_{\beta_{3+M(2MF+1)}}^k}{\rho} - v_{\beta_{3+M(2MF+1)}}^k \right\|^2, \tag{9.32a}$$

where, $\beta_1 = \beta(j, j)$, $\beta_2 = \beta(i + MF(M-1), i)$ and $\beta_3 = \beta(i + MF(M-1) + M, i)$.

Using the previous simplifications obtained on the auxiliary variables and the Lagrangian multipliers, we can solve the primal variables update steps.

[2] i does not really point to the right constraint in matrix $\mathbf{A}$ because we made a transition in order to follow the entries of the corresponding slack vector.

The final equations for each SBS j are as follows:

$$x_{j0f}^{k+1} = \left[x_{j0f}^{k} - \left(\frac{\pi_{j_{+MF(M-1)+M+f)}}}{\rho} + \frac{r_{j_{+MF(M-1)+M+f}}(\boldsymbol{u}^k)}{d(j+MF(M-1)+M+f)} + \frac{c_{j0}s_f d_{jf}}{\rho} \right) \right]_+ \tag{9.33a}$$

$$x_{jjf}^{k+1} = \left[x_{jjf}^{k} - \frac{1}{1+s_f^2} \left(\frac{s_f \pi_{j_{+MF(M-1)}}}{\rho} + \frac{s_f r_{j_{+MF(M-1)}}(\boldsymbol{u}^k)}{d(j+MF(M-1))} + \right. \right.$$
$$\left. \left. \frac{\pi_{j_{+MF(M-1)+M+f}}}{\rho} + \frac{r_{j_{+MF(M-1)+M+f}}(\boldsymbol{u}^k)}{d(j+MF(M-1)+M+f)} + \frac{c_{j0}s_f}{\rho} \right) \right]_+ \tag{9.33b}$$

$$x_{jj'f}^{k+1} = \left[x_{jj'f}^{k} - \frac{1}{\rho} \left(\pi^k_{(M-1)F(j-1)+F(j'-1-1_{j'>j})+f} + \right. \right.$$
$$\frac{\rho r_{(M-1)F(j-1)+F(j'-1-1_{j'>j})+f}(\boldsymbol{u}^k)}{d((M-1)F(j-1)+F(j'-1-1_{j'>j})+f)} + \pi_{F(j-1)+MF(M-1)+M+f} +$$
$$\left. \left. \frac{\rho r_{F(j-1)+MF(M-1)+M+f}(\boldsymbol{u}^k)}{d(F(j-1)+MF(M-1)+M+f)} + c_{jj'}s_f d_{jf} \right) \right]_+ \tag{9.33c}$$

$$sl_{1,i}^{k+1} = \left[sl_{1,i}^{k} - \frac{\pi_i^k}{\rho} - \frac{r_i(\boldsymbol{u}^k)}{d(i)} \right]_+ \tag{9.33d}$$

$$sl_{2,i}^{k+1} = \left[sl_{2,i}^{k} - \frac{\pi^k_{i_{+MF(M-1)}}}{\rho} - \frac{r_{i_{+MF(M-1)}}(\boldsymbol{u}^k)}{d(i+MF(M-1))} \right]_+ \tag{9.33e}$$

$$sl_{3,i}^{k+1} = \left[sl_{3,i}^{k} - \frac{\pi^k_{i_{+MF(M-1)+M}}}{\rho} - \frac{r_{i_{+MF(M-1)+M}}(\boldsymbol{u}^k)}{d(i+MF(M-1)+M)} \right]_+ \tag{9.33f}$$

$$\tag{9.33g}$$

We regroup all the final update steps. These updates can be applied synchronously or asynchronously. When applied in a synchronous manner, we obtain algorithm 8 that summarizes the distributed caching with sharing obtained using ADMM. When implemented asynchronously, the update steps have to be applied by the node or set of nodes that were switched *on*.

Algorithm 8: Distributed Caching while sharing w/ADMM

1. Initialize the primal & slack variables, and the Lagrangian multipliers to a desired value.
2. At iteration $k+1$:
   (a) Every SBS $j = 1, ..., M$ decides whether to cache a content or not and if it wants to share it by updating its variables using Eq. (9.33a) to (9.33f)
   (b) Exchange the values corresponding to the cached files between the connected SBSs. I.e., SBS j communicate $x_{jjf}^{k+1}, \forall f = 1, \ldots, F.$ to its neighbors.
   (c) Every SBS $j = 1, ..., M$ updates the Lagrangian multipliers corresponding to its constraints using

$$\pi_i^{k+1} = \pi_i^k + \frac{\rho}{d(i)} r_i(\boldsymbol{u}^{k+1})$$

3. Each SBS checks if its constraints are violated then increase k and return to step 2. Otherwise, SBS j stores and shares the files as given by its primal variables vector.

9.6 Conclusion

In this chapter we provided a distributed ADMM approach to solve the caching problem in $5G$ networks. We introduced the file sharing concept between the nodes and considered different sizes for each file. Thus, the nodes have an additional choice that they can privilege. They can chose to share files among each other, and not only by fetching it from the CS. We provided the detailed steps to obtain the distributed updates to be applied by each node. These update steps can be performed by all the nodes within an iteration in a synchronous fashion. Additionally, the nodes can choose not to update their variables, a situation in which the ADMM algorithm is applied asynchronously. In both cases the algorithm inherent the convergence property detailed in the previous chapters. The ADMM application in such context does always converge, an optimal solution minimizing the global cost is thus obtained.

Chapter 10

Conclusions

10.1 Summary

The work carried out in this dissertation focused on distributed synchronous and asynchronous Alternating Direction method of Multipliers (ADMM) applications in large systems of interconnected nodes. Depending on the analyzed system, different algorithms were provided and explained.

In the first part of this thesis we were concerned about the smart energy management of the power grid.

In Chapter 3 we gave a detailed study on the power grid progress towards becoming smarter and the required interventions at different levels. We placed ourselves in the management part, specifically on the problem related to optimal power flow (OPF) and its variant the Direct-Current optimal power flow (DC-OPF). We introduced some of the methods that can be used and their applications. Additionally, we highlighted the related works achievements and drawbacks.

In Chapter 4, we focused on the DC-OPF problem formulation as a Production-Sharing problem. This formulation was used in order to show how the work provided in this thesis can cope with various systems and problems. At this point of the thesis, we sought to solve distributively the Production-Sharing problem. Drawn by the performance limitations of the existing methods, we turned our attention to the Alternation Direction Method of Multipliers (ADMM) in Chapter 5. ADMM has been proved to perform very well for distributed optimization over interconnected networks. Using some of the basic monotone operator theory notions on the dual of the Distributed Production-Sharing (DPS) problem, we formulated an area-based version of the ADMM. We also explained the process of obtaining the classical ADMM version using the same procedure. The area-based ADMM

algorithm was obtained by combining the Douglas-Rachford (DR) splitting method and the Proximal Point algorithm (PPA). DR splitting method was used to split the dual of the problem that can be seen as the sum of two maximal monotone operators. From these operators we obtained the DR operator. Then, PPA was applied on the DR operator, leading to the area-based distributed ADMM.

We implemented this method on the DPS problem in Chapter 6. The direct application tends to inherit the same synchronous limitation of the existing methods. Using random Gauss-Seidel iterations on PPA we provided an asynchronous version of the ADMM method. The convergence of both versions was limited to the case of overlapping areas. In power grid networks this situation may not be feasible. For this reason, we introduced dummy nodes on the tie-lines connecting the nodes belonging to different areas. This helped us in extending the application of the synchronous and asynchronous ADMM algorithms to the case where the areas do not overlap.

In Chapter 7 we introduced renewable energy sources and storage devices to the conventional power grid. Such equipment provided the network with the ability of sharing energy across time while producing green cheap power. After describing the model in hand we formulated the problem and we solved it by combining Lyapunov techniques and ADMM. Lyapunov techniques were used to design a low complexity online solution of the problem after relaxing the storage devices constraints. Perturbation parameters were introduced. These parameters were tuned so that the relaxed problem solution is also feasible for the original problem. Finally, a distributed implementation of the online algorithm was provided using ADMM.

In Chapter 8 we implemented the ADMM algorithms on different networks. We used the IEEE−30 and 118 Bus test systems for the implementation of the DC-OPF problem in the case of conventional power grid. We compared the objective convergence and residual convergence of the algorithms under multiple scenarios. The algorithms tend to converge under all circumstances. An additional number of iterations was required to obtain the same result when going from the overlapping to the non-overlapping architectures. We also confirmed through these simulations the convergence of the asynchronous area-based ADMM and its scalability when the network tends to become larger. Even when a big portion of the network is failing or not updating its variables, we were able to attain the optimal solution. Finally, for the case of a network with distributed generation units and storage devices, we used the IEEE−6 Bus test system. Due to the presence of storage devices, we were able to observe the effect of of sharing across time. The storage devices helped in reducing the global cost across time when we increased their charging/discharging abilities.

In the second part of this dissertation, we gave our initial results obtained by using ADMM for distributed Caching in 5G Networks

In Chapter 9 after introducing the problem, we cited some of the related work in this field. We explained the network model and formulated the caching problem as a convex optimization problem. Conventionally the central scheduler had to solve this problem in a centralized fashion. Thus, it is should be endowed large computational capacities. Communications had to be done between all the nodes and this scheduler. For these reasons, we reformulated the problem and solved it distributively at node level using ADMM. In Chapter 9 we supposed that the nodes were allowed to share contents. We provided an example of a network of 3 nodes to illustrate the problem. We detailed how to obtain an ADMM algorithm for a more general network with arbitrary number of nodes and contents. From this formulation, we are more convinced that the distributed optimization using ADMM is really appealing for such an area of application. This is mainly due to the fact that the problem size scales with the number of nodes, connections and contents.

10.2 Directions for Future Works

One of the main challenges is implementing the distributed ADMM algorithms, which we have derived in this thesis, on a real network with real data measurements. This allows investigating the behavior of the network under node failure or area failure in comparison with the synchronous implementation of the distributed ADMM algorithm when no failures occur. This also helps in examining the scalability of the algorithms when the network tends to become of large scale.

Another challenge is to accelerate the convergence speeds. We usually need several hundreds of iterations to achieve the convergence. A better step-adaptation procedure may help in decreasing the required number of iterations. In fact, we do not have yet a direct relation between the parameter ρ (which imposes a penalty to violating the constraints) and the convergence speed of the distributed ADMM applied to the DC-OPF problem. We tried to find such relation by studying the convergence of the algorithm and its dependency with the penalty parameter ρ. First, we progressed by finding the matrix relating the zeros of the Douglas-Rachford splitting operator ζ^{k+1} and ζ^{k}. Then, we needed to proceed with an eigenvalue decomposition and a formal study of this matrix eigenvalues. However, this decomposition required lots of effort to be obtained even for a small simple network. Proceeding in this study may lead to a significant enhancement in the convergence speed.

Node, link or area failure poses a crucial problem in many actual domains. The asynchronous distributed ADMM algorithm considers this case, where the area update process is chosen to be i.i.d. Besides, we can study the case where the random area selection has a Markovian structure. If an area, a link, or a node is off, most likely it will still be off on the next round and hence the random area selection has a Markovian structure. We need to investigate in details such case that tends towards a more realistic implementation of the network with failures case.

One may also focus on using the asynchronous distributed optimization using ADMM when the distributed generation (DG) sources are integrated into the network and the nodes are equipped with storage devices. After completing the work of Chapter 7, the authors of [111] published similar results where they combined Lyapunov optimization with ADMM. This limited our ability to publish our work related to this application. In addition, we started a joint work on the robustness of the distributed ADMM algorithm when the network faces failures or attacks. We need to study the behavior of the algorithm when load redistribution attacks exist and the effect that distributed storage devices can induce in such cases. When a load attack occurs and may lead to shedding, we need then to study the usage of storage devices to cover such an unfulfilled demand.

Regarding the distributed caching in 5G networks, the next step is to take into account the users when formulating the network model in the caching problem. This will add another level in the caching vector of each node. We then need to consider clustering the users and how these users should be grouped while being constrained with the nodes available power.

Lastly, combining the two parts studied in this dissertation, i.e. energy efficient caching using DG units power excess, can also be of a great interest for future studies. We need to investigate the usage of the extra generation from these sources to power the SBS and make them cache the files efficiently.

Appendix A

Data tables for IEEE Test bus systems

Table A.1: Transmission Lines data, IEEE−30 Bus Test System

Line	**from node** i	**to node** j	r_{ij}	x_{ij}	b_{ij}	$\underline{p_{ij}}$	$\overline{p_{ij}}$
1	1	2	0,0192	0,0575	0,0528	-200	200
2	1	3	0,0452	0,1652	0,0408	-200	200
3	2	4	0,057	0,1737	0,0368	-200	200
4	3	4	0,0132	0,0379	0,0084	-200	200
5	2	5	0,0472	0,1983	0,0418	-200	200
6	2	6	0,0581	0,1763	0,0374	-200	200
7	4	6	0,0119	0,0414	0,009	-200	200
8	5	7	0,046	0,116	0,0204	-200	200
9	6	7	0,0267	0,082	0,017	-200	200
10	6	8	0,012	0,042	0,009	-200	200
11	6	9	0	0,208	0	-200	200
12	6	10	0	0,556	0	-200	200
13	9	11	0	0,208	0	-200	200
14	9	10	0	0,11	0	-200	200
15	4	12	0	0,256	0	-200	200
16	12	13	0	0,14	0	-200	200

17	12	14	0,1231	0,2559	0	-200	200
18	12	15	0,0662	0,1304	0	-200	200
19	12	16	0,0945	0,1987	0	-200	200
20	14	15	0,221	0,1997	0	-200	200
21	16	17	0,0524	0,1923	0	-200	200
22	15	18	0,1073	0,2185	0	-200	200
23	18	19	0,0639	0,1292	0	-200	200
24	19	20	0,034	0,068	0	-200	200
25	10	20	0,0936	0,209	0	-200	200
26	10	17	0,0324	0,0845	0	-200	200
27	10	21	0,0348	0,0749	0	-200	200
28	10	22	0,0727	0,1499	0	-200	200
29	21	22	0,0116	0,0236	0	-200	200
30	15	23	0,1	0,202	0	-200	200
31	22	24	0,115	0,179	0	-200	200
32	23	24	0,132	0,27	0	-200	200
33	24	25	0,1885	0,3292	0	-200	200
34	25	26	0,2544	0,38	0	-200	200
35	25	27	0,1093	0,2087	0	-200	200
36	28	27	0	0,396	0	-200	200
37	27	29	0,2198	0,4153	0	-200	200
38	27	30	0,3202	0,6027	0	-200	200
39	29	30	0,2399	0,4533	0	-200	200
40	8	28	0,0636	0,2	0,0428	-200	200
41	6	28	0,0169	0,0599	0,013	-200	200

Table A.2: Nodes data, IEEE−118 Bus Test System

Node j	c_j	$\underline{p_j^G}$	$\overline{p_j^G}$	p_j^D	c'_j	Node j	c_j	$\underline{p_j^G}$	$\overline{p_j^G}$	p_j^D	c'_j
1	40	5	30	51	0	60	0	0	0	78	0
2	0	0	0	20	0	61	20	50	200	0	0,07
3	0	0	0	39	0	62	40	25	100	77	0,09
4	40	5	30	39	0,1	63	0	0	0	0	0
5	0	0	0	0	0	64	0	0	0	0	0
6	40	5	30	52	0,07	65	20	100	420	0	0,07
7	0	0	0	19	0	66	20	100	420	39	0,14
8	40	150	300	28	0,08	67	0	0	0	28	0
9	0	0	0	0	0	68	0	0	0	0	0
10	20	100	300	0	0,04	69	20	80	300	0	0,1
11	0	0	0	70	0	70	40	30	80	66	0,09
12	20	10	30	47	0,01	71	0	0	0	0	0
13	0	0	0	34	0	72	40	10	30	12	0,04
14	0	0	0	14	0	73	40	5	30	6	0,08
15	40	25	100	90	0,05	74	40	5	20	68	0,13
16	0	0	0	25	0	75	0	0	0	47	0
17	0	0	0	11	0	76	40	25	100	68	0,1
18	40	5	30	60	0.1	77	40	25	100	61	0,06
19	40	5	30	45	0.04	78	0	0	0	71	0
20	0	0	0	18	0	79	0	0	0	39	0
21	0	0	0	14	0	80	20	150	300	130	0,06
22	0	0	0	10	0	81	0	0	0	0	0
23	0	0	0	7	0	82	0	0	0	54	0,02
24	40	100	300	13	0.05	83	0	0	0	20	0
25	20	100	350	0	0.11	84	0	0	0	11	0
26	20	8	30	0	0,04	85	40	25	100	24	0,06
27	40	8	30	71	0,1	86	0	0	0	21	0
28	0	0	0	17	0	87	20	10	30	0	0,1
29	0	0	0	24	0	88	0	0	0	48	0
30	0	0	0	0	0	89	20	100	300	0	0,08

31	20	25	100	43	0,05	90	40	50	200	163	0,07
32	40	8	30	59	0,13	91	40	8	20	10	0,06
33	0	0	0	23	0	92	40	20	50	65	0,09
34	40	25	100	59	0,07	93	0	0	0	12	0
35	0	0	0	33	0	94	0	0	0	30	0
36	40	8	30	31	0,04	95	0	0	0	42	0
37	0	0	0	0	0	96	0	0	0	38	0
38	0	0	0	0	0	97	0	0	0	15	0
39	0	0	0	27	0	98	0	0	0	34	0
40	40	8	30	66	0,05	99	40	100	300	42	0,03
41	0	0	0	37	0	100	20	100	300	37	0,04
42	40	25	100	96	0,1	101	0	0	0	22	0
43	0	0	0	18	0	102	0	0	0	5	0
44	0	0	0	16	0	103	20	100	300	23	0,05
45	0	0	0	53	0	104	40	8	20	38	0,06
46	20	50	250	28	0,07	105	40	25	100	31	0,07
47	0	0	0	34	0	106	0	0	0	43	0
48	0	0	0	20	0	107	40	25	100	50	0,03
49	20	50	250	87	0,08	108	0	0	0	2	0
50	0	0	0	17	0	109	0	0	0	8	0
51	0	0	0	17	0	110	40	8	20	39	0,04
52	0	0	0	18	0	111	20	25	50	0	0,04
53	0	0	0	23	0	112	40	25	100	68	0,08
54	20	25	100	113	0,08	113	40	25	100	6	0,09
55	40	25	100	63	0,05	114	0	0	0	8	0
56	40	50	200	84	0,01	115	0	0	0	22	0
57	0	0	0	12	0	116	40	25	100	184	0,07
58	0	0	0	12	0	117	0	0	0	20	0
59	10	0	100	277	0,01	118	0	0	0	33	0

Bibliography

[1] A. J. Wood and B. F. Wollenberg, *Power generation, operation, and control*. John Wiley & Sons, 2012.
(cited in pages 2, 3, 15, 16, 17, 68, 112)

[2] A. Gómez-Expósito, A. J. Conejo, and C. Cañizares, *Electric energy systems: analysis and operation*. CRC Press, 2008.
(cited in pages 2, 15, 16, 27)

[3] ENTSO-E vision: The energy union aheadreliable, sustainable, competitive power systems: https://www.entsoe.eu/pages/default.aspx.
(cited in pages 2, 15)

[4] F. C. Schweppe and W. J, "Power system static-state estimation, part i: Exact model," *power apparatus and systems, ieee transactions on*, no. 1, pp. 120–125, 1970.
(cited in pages 3, 17)

[5] F. C. Schweppe and D. B. Rom, "Power system static-state estimation, part ii: Approximate model," *power apparatus and systems, ieee transactions on*, no. 1, pp. 125–130, 1970.
(cited in pages 3, 17)

[6] F. C. Schweppe, "Power system static-state estimation, part iii: Implementation," *Power Apparatus and Systems, IEEE Transactions on*, no. 1, pp. 130–135, 1970.
(cited in pages 3, 17)

[7] F. F. Wu, K. Moslehi, and A. Bose, "Power system control centers: Past, present, and future," *Proceedings of the IEEE*, vol. 93, no. 11, pp. 1890–1908, 2005.
(cited in pages 3, 17)

[8] S. Kar, G. Hug, J. Mohammadi, and J. M. Moura, "Distributed state estimation and energy management in smart grids: A consensus inno-

vations approach," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 8, no. 6, pp. 1022–1038, 2014.
(cited in pages 3, 17)

[9] U. Congress, "Energy policy act of 2005," *Public Law*, vol. 109, no. 58, p. 42, 2005.
(cited in pages 3, 17)

[10] B. H. Chowdhury and S. Rahman, "A review of recent advances in economic dispatch." Institute of Electrical and Electronics Engineers, 1990.
(cited in pages 3, 17)

[11] R. Burchett, H. H. Happ, D. Vierath, and K. Wirgau, "Developments in optimal power flow," *Power Apparatus and Systems, IEEE Transactions on*, no. 2, pp. 406–414, 1982.
(cited in pages 4, 18)

[12] O. Alsac and B. Stott, "Optimal load flow with steady-state security," *Power Apparatus and Systems, IEEE Transactions on*, no. 3, pp. 745–751, 1974.
(cited in pages 4, 18)

[13] M. Osman, M. A. Abo-Sinna, and A. Mousa, "A solution to the optimal power flow using genetic algorithm," *Applied mathematics and computation*, vol. 155, no. 2, pp. 391–405, 2004.
(cited in pages 4, 18)

[14] B. Stott, J. Jardim, and O. Alsac, "DC power flow revisited," *IEEE Transactions on Power Systems*, vol. 24, no. 3, pp. 1290–1300, 2009.
(cited in pages 4, 18)

[15] F. Li and R. Bo, "Dcopf-based lmp simulation: algorithm, comparison with acopf, and sensitivity," *Power Systems, IEEE Transactions on*, vol. 22, no. 4, pp. 1475–1485, 2007.
(cited in pages 4, 18)

[16] M. Davari, F. Toorani, H. Nafisi, M. Abedi, and G. Gharepatian, "Determination of mean and variance of lmp using probabilistic dcopf and t-pem," *PECon08. pp*, pp. 1280–1283, 2008.
(cited in pages 4, 18)

[17] R. D. Christie, B. F. Wollenberg, and I. Wangensteen, "Transmission management in the deregulated environment," *Proceedings of the*

IEEE, vol. 88, no. 2, pp. 170–195, 2000.
(cited in pages 4, 18)

[18] B. Stott, O. Alsac, and A. J. Monticelli, "Security analysis and optimization," *Proceedings of the IEEE*, vol. 75, no. 12, pp. 1623–1644, 1987.
(cited in pages 4, 19)

[19] Cisco, "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2014-2019," *White Paper, [Online] http://goo.gl/k84Qpo*, 2015.
(cited in pages 5, 117)

[20] J. G. Andrews, S. Buzzi, W. Choi, S. Hanly, A. Lozano, A. C. Soong, and J. C. Zhang, "What will 5G be?" *arXiv preprint: 1405.2957*, 2014.
(cited in pages 5, 117)

[21] S. Massoud Amin and B. F. Wollenberg, "Toward a smart grid: power delivery for the 21st century," *IEEE Power and Energy Magazine*, vol. 3, no. 5, pp. 34–41, 2005.
(cited in pages 16, 31)

[22] H. Farhangi, "The path of the smart grid," *Power and Energy Magazine, IEEE*, vol. 8, no. 1, pp. 18–28, 2010.
(cited in page 16)

[23] Y. Yan, Y. Qian, H. Sharif, and D. Tipper, "A survey on smart grid communication infrastructures: Motivations, requirements and challenges," *Communications Surveys & Tutorials, IEEE*, vol. 15, no. 1, pp. 5–20, 2013.
(cited in page 16)

[24] X. Fang, S. Misra, G. Xue, and D. Yang, "Smart gridthe new and improved power grid: A survey," *Communications Surveys & Tutorials, IEEE*, vol. 14, no. 4, pp. 944–980, 2012.
(cited in page 16)

[25] T. H. Kjeldsen, "A contextualized historical analysis of the kuhn–tucker theorem in nonlinear programming: The impact of world war ii," *Historia mathematica*, vol. 27, no. 4, pp. 331–361, 2000.
(cited in page 18)

[26] H. W. Kuhn, "Nonlinear programming: a historical view," in *Traces and Emergence of Nonlinear Programming*. Springer, 2014, pp. 393–414.
(cited in page 18)

[27] W. Karush, *Minima of functions of several variables with inequalities as side conditions*. Springer, 2014.
(cited in page 18)

[28] K. Pandya and S. Joshi, "A survey of optimal power flow methods." *Journal of Theoretical & Applied Information Technology*, vol. 4, no. 5, 2008.
(cited in page 18)

[29] M. Huneault, F. Galiana, and Q. St Bruno, "A survey of the optrial power flow literature," *IEEE Transactions on Power Systems*, vol. 6, no. 2, 1991.
(cited in page 18)

[30] J. A. Momoh, M. El-Hawary, and R. Adapa, "A review of selected optimal power flow literature to 1993. part ii: Newton, linear programming and interior point methods," *IEEE Transactions on Power Systems*, vol. 14, no. 1, pp. 105–111, 1999.
(cited in pages 18, 19)

[31] ——, "A review of selected optimal power flow literature to 1993. part i: Nonlinear and quadratic programming approaches," *IEEE transactions on power systems*, vol. 14, no. 1, pp. 96–104, 1999.
(cited in pages 18, 19)

[32] D. P. Bertsekas, "Nonlinear programming," 1999.
(cited in page 19)

[33] A. Wolf, J. B. Swift, H. L. Swinney, and J. A. Vastano, "Determining lyapunov exponents from a time series," *Physica D: Nonlinear Phenomena*, vol. 16, no. 3, pp. 285–317, 1985.
(cited in page 19)

[34] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.
(cited in page 19)

[35] G. B. Dantzig, *Linear programming and extensions*. Princeton university press, 1998.
(cited in page 19)

[36] W. L. Winston and J. B. Goldberg, *Operations research: applications and algorithms*. Duxbury press Boston, 2004, vol. 3.
(cited in page 19)

[37] P. T. Boggs and J. W. Tolle, "Sequential quadratic programming," *Acta numerica*, vol. 4, pp. 1–51, 1995.
(cited in page 19)

[38] S. Frank, I. Steponavice, and S. Rebennack, "Optimal power flow: a bibliographic survey i," *Energy Systems*, vol. 3, no. 3, pp. 221–258, 2012.
(cited in page 19)

[39] R. Bansal, "Bibliography on the fuzzy set theory applications in power systems (1994-2001)," *Power Systems, IEEE Transactions on*, vol. 18, no. 4, pp. 1291–1299, 2003.
(cited in page 19)

[40] S. Li, T. Haskew, L. Xu *et al.*, "Control of hvdc light system using conventional and direct current vector control approaches," *Power Electronics, IEEE Transactions on*, vol. 25, no. 12, pp. 3106–3118, 2010.
(cited in page 19)

[41] V. Ramesh and X. Li, "A fuzzy multiobjective approach to contingency constrained opf," *Power Systems, IEEE Transactions on*, vol. 12, no. 3, pp. 1348–1354, 1997.
(cited in page 19)

[42] G. Celli, F. Pilo, G. Pisano, and G. Soma, "Optimal participation of a microgrid to the energy market with an intelligent ems," in *Power Engineering Conference, 2005. IPEC 2005. The 7th International*. IEEE, 2005, pp. 663–668.
(cited in page 19)

[43] R. Azami, M. S. Javadi, and G. Hematipour, "Economic load dispatch and dc-optimal power flow problem-pso versus lr," *Int. J. Multidscip. Sci. Eng*, vol. 2, no. 9, pp. 8–13, 2011.
(cited in page 19)

[44] R. C. Eberhart and Y. Shi, "Particle swarm optimization: developments, applications and resources," in *Evolutionary Computation, 2001. Proceedings of the 2001 Congress on*, vol. 1. IEEE, 2001, pp. 81–86.
(cited in page 19)

[45] B. Zhao, C. Guo, and Y. Cao, "A multiagent-based particle swarm optimization approach for optimal reactive power dispatch," *Power Systems, IEEE Transactions on*, vol. 20, no. 2, pp. 1070–1078, 2005.
(cited in page 19)

[46] H. Yoshida, K. Kawata, Y. Fukuyama, S. Takayama, and Y. Nakanishi, "A particle swarm optimization for reactive power and voltage control considering voltage security assessment," *Power Systems, IEEE Transactions on*, vol. 15, no. 4, pp. 1232–1239, 2000.
(cited in page 19)

[47] M. Abido, "Optimal power flow using particle swarm optimization," *International Journal of Electrical Power & Energy Systems*, vol. 24, no. 7, pp. 563–571, 2002.
(cited in page 19)

[48] J. Li, C.-C. Liu, and K. P. Schneider, "Controlled partitioning of a power network considering real and reactive power balance," *Smart Grid, IEEE Transactions on*, vol. 1, no. 3, pp. 261–269, 2010.
(cited in page 20)

[49] D. P. Bertsekas, "Multiplier methods: a survey," *Automatica*, vol. 12, no. 2, pp. 133–145, 1976.
(cited in page 21)

[50] R. T. Rockafellar, "A dual approach to solving nonlinear programming problems by unconstrained optimization," *Mathematical Programming*, vol. 5, no. 1, pp. 354–373, 1973.
(cited in page 21)

[51] J. Batut and A. Renaud, "Daily generation scheduling optimization with transmission constraints: a new class of algorithms," *Power Systems, IEEE Transactions on*, vol. 7, no. 3, pp. 982–989, 1992.
(cited in page 21)

[52] B. H. Kim and R. Baldick, "Coarse-grained distributed optimal power flow," *IEEE Transactions on Power Systems*, vol. 12, no. 2, pp. 932–

939, 1997.
(cited in page 21)

[53] G. Cohen, "Auxiliary problem principle and decomposition of optimization problems," *Journal of optimization Theory and Applications*, vol. 32, no. 3, pp. 277–305, 1980.
(cited in page 21)

[54] ——, "Optimization by decomposition and coordination: a unified approach," *Automatic Control, IEEE Transactions on*, vol. 23, no. 2, pp. 222–232, 1978.
(cited in page 21)

[55] B. H. Kim and R. Baldick, "A comparison of distributed optimal power flow algorithms," *Power Systems, IEEE Transactions on*, vol. 15, no. 2, pp. 599–604, 2000.
(cited in page 21)

[56] D. G. Luenberger, *Linear and nonlinear programming*. Springer, 2003.
(cited in page 21)

[57] A. J. Conejo, F. J. Nogales, and F. J. Prieto, "A decomposition procedure based on approximate newton directions," *Mathematical programming*, vol. 93, no. 3, pp. 495–515, 2002.
(cited in page 21)

[58] T. Erseghe, "Distributed optimal power flow using admm," *Power Systems, IEEE Transactions on*, vol. 29, no. 5, pp. 2370–2380, 2014.
(cited in pages 21, 22)

[59] M. Coelho, A. Oliveira, and S. Soares, "Optimal power flow in dc constrained operating reserve through interior point methods," in *2012 Sixth IEEE/PES Transmission and Distribution: Latin America Conference and Exposition (T&D-LA)*, 2012.
(cited in page 21)

[60] S. Granville, "Optimal reactive dispatch through interior point methods," *Power Systems, IEEE Transactions on*, vol. 9, no. 1, pp. 136–146, 1994.
(cited in page 21)

[61] V. H. Quintana, G. L. Torres, and J. Medina-Palomo, "Interior-point methods and their applications to power systems: a classification of publications and software codes," *Power Systems, IEEE Transactions*

on, vol. 15, no. 1, pp. 170–176, 2000.
(cited in page 21)

[62] K. Deb, *Optimization for engineering design: Algorithms and examples*. PHI Learning Pvt. Ltd., 2012.
(cited in page 21)

[63] S. Paudyal, C. Canizares, K. Bhattacharya *et al.*, "Three-phase distribution opf in smart grids: Optimality versus computational burden," in *Innovative Smart Grid Technologies (ISGT Europe), 2011 2nd IEEE PES International Conference and Exhibition on*. IEEE, 2011, pp. 1–7.
(cited in page 21)

[64] H. Kung, "Synchronized and asynchronous parallel algorithms for multiprocessors," 1976.
(cited in page 21)

[65] R. Glowinski and A. Marroco, "Sur l'approximation, par éléments finis d'ordre un, et la résolution, par pénalisation-dualité d'une classe de problèmes de dirichlet non linéaires," *Revue française d'automatique, informatique, recherche opérationnelle. Analyse numérique*, vol. 9, no. 2, pp. 41–76, 1975.
(cited in page 21)

[66] D. Gabay and B. Mercier, "A dual algorithm for the solution of nonlinear variational problems via finite element approximation," *Computers & Mathematics with Applications*, vol. 2, no. 1, pp. 17–40, 1976.
(cited in page 21)

[67] J. Eckstein, "Parallel alternating direction multiplier decomposition of convex programs," *Journal of Optimization Theory and Applications*, vol. 80, no. 1, pp. 39–62, 1994.
(cited in page 21)

[68] J. Eckstein and D. P. Bertsekas, "On the Douglas–Rachford splitting method and the proximal point algorithm for maximal monotone operators," *Mathematical Programming*, vol. 55, no. 1-3, pp. 293–318, 1992.
(cited in pages 21, 22, 36, 39, 40, 43, 48)

[69] J. Eckstein, "Alternating Direction Multiplier Decomposition of Convex Problems," 1991.
(cited in pages 21, 35)

[70] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.
(cited in pages 21, 22, 25, 118)

[71] V. Kekatos and G. Giannakis, "Distributed robust power system state estimation," *Power Systems, IEEE Transactions on*, vol. 28, no. 2, pp. 1617–1626, 2013.
(cited in page 22)

[72] S. Magnússon, P. Weeraddana, and C. Fischione, "A distributed approach for the optimal power flow problem based on admm and sequential convex approximations," 2014.
(cited in page 22)

[73] Q. Peng and S. H. Low, "Distributed algorithm for optimal power flow on a radial network," in *Decision and Control (CDC), 2014 IEEE 53rd Annual Conference on*. IEEE, 2014, pp. 167–172.
(cited in page 22)

[74] P. Sulc, S. Backhaus, and M. Chertkov, "Optimal distributed control of reactive power via the alternating direction method of multipliers," *Energy Conversion, IEEE Transactions on*, vol. 29, no. 4, pp. 968–977, 2014.
(cited in page 22)

[75] J. Rivera, P. Wolfrum, S. Hirche, C. Goebel, and H.-A. Jacobsen, "Alternating direction method of multipliers for decentralized electric vehicle charging control," in *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*. IEEE, 2013, pp. 6960–6965.
(cited in page 22)

[76] E. Dall'Anese, H. Zhu, and G. Giannakis, "Distributed optimal power flow for smart microgrids," *Smart Grid, IEEE Transactions on*, vol. 4, no. 3, pp. 1464–1475, 2013.
(cited in page 22)

[77] F. Iutzeler, P. Bianchi, P. Ciblat, and W. Hachem, "Asynchronous Distributed Optimization using a Randomized Alternating Direction Method of Multipliers," *IEEE Conference on Decision and Control (CDC)*, 2013.
(cited in pages 22, 35, 59, 60)

[78] L. Powell, *Power System Load Flow Analysis (Professional Engineering)*. McGraw-Hill Professional, 2004.
(cited in page 27)

[79] J. D. Glover, M. S. Sarma, and T. J. Overbye, *Power system analysis and design*. CengageBrain. com, 2011.
(cited in pages 27, 79)

[80] N. S. Rau, *Optimization principles: practical applications to the operation and markets of the electric power industry*. John Wiley & Sons, 2003, vol. 16.
(cited in page 28)

[81] A. G. Bakirtzis and P. N. Biskas, "A decentralized solution to the dc-opf of interconnected power systems," *Power Systems, IEEE Transactions on*, vol. 18, no. 3, pp. 1007–1013, 2003.
(cited in page 28)

[82] M. Kraning, E. Chu, J. Lavaei, and S. Boyd, "Dynamic network energy management via proximal message passing," *Foundations and Trends in Optimization*, vol. 1, no. 2, pp. 73–126, 2014.
(cited in page 31)

[83] R. T. Rockafellar, "Monotone operators and the proximal point algorithm," *SIAM Journal on Control and Optimization*, vol. 14, no. 5, pp. 877–898, 1976.
(cited in pages 36, 38)

[84] G. J. Minty *et al.*, "On the maximal domain of a"monotone"function." *The Michigan Mathematical Journal*, vol. 8, no. 2, pp. 135–137, 1961.
(cited in page 36)

[85] R. T. Rockafellar, "On the maximal monotonicity of subdifferential mappings," *Pacific J. Math*, vol. 33, no. 1, pp. 209–216, 1970.
(cited in pages 36, 38, 39)

[86] H. Bauschke and P. Combettes, *Convex analysis and monotone operator theory in Hilbert spaces*. Springer, 2011.
(cited in pages 37, 38, 41, 44)

[87] J. Eckstein, "Splitting methods for monotone operators with applications to parallel optimization," Ph.D. dissertation, Massachusetts Institute of Technology, 1989.
(cited in page 38)

[88] P. L. Combettes and J.-C. Pesquet, "Proximal splitting methods in signal processing," in *Fixed-point algorithms for inverse problems in science and engineering*. Springer, 2011, pp. 185–212.
(cited in page 38)

[89] J. Douglas and H. H. Rachford, "On the numerical solution of heat conduction problems in two and three space variables," *Transactions of the American mathematical Society*, pp. 421–439, 1956.
(cited in page 39)

[90] P.-L. Lions and B. Mercier, "Splitting algorithms for the sum of two nonlinear operators," *SIAM Journal on Numerical Analysis*, vol. 16, no. 6, pp. 964–979, 1979.
(cited in page 39)

[91] I. D. Schizas, A. Ribeiro, and G. B. Giannakis, "Consensus in ad hoc wsns with noisy linkspart i: Distributed estimation of deterministic signals," *IEEE Transactions on Signal Processing*, vol. 56, no. 1, pp. 350–364, 2008.
(cited in page 56)

[92] P. Bianchi, W. Hachem, and F. Iutzeler, "A stochastic coordinate descent primal-dual algorithm and applications to large-scale composite optimization," *arXiv preprint arXiv:1407.0898*, 2014.
(cited in page 59)

[93] A. J. Conejo and J. A. Aguado, "Multi-area coordinated decentralized DC optimal power flow," *IEEE Transactions on Power Systems*, vol. 13, no. 4, pp. 1272–1278, 1998.
(cited in pages 62, 63)

[94] L. Georgiadis, M. J. Neely, and L. Tassiulas, *Resource allocation and cross-layer control in wireless networks*. Now Publishers Inc, 2006.
(cited in pages 67, 70)

[95] S. Lakshminarayana, T. Q. Quek, and H. V. Poor, "Cooperation and storage tradeoffs in power grids with renewable energy resources," *Selected Areas in Communications, IEEE Journal on*, vol. 32, no. 7, pp. 1386–1397, 2014.
(cited in pages 67, 70, 71, 72)

[96] R. Urgaonkar, B. Urgaonkar, M. J. Neely, and A. Sivasubramaniam, "Optimal power cost management using stored energy in data centers,"

in *Proceedings of the ACM SIGMETRICS joint international conference on Measurement and modeling of computer systems*. ACM, 2011, pp. 221–232.
(cited in page 70)

[97] IEEE power systems: http://www.ee.washington.edu/research/pstca.
(cited in pages 79, 106, 115)

[98] E. Baştuğ, M. Bennis, and M. Debbah, "Living on the Edge: The role of proactive caching in 5G wireless networks," *IEEE Communications Magazine*, vol. 52, no. 8, pp. 82–89, August 2014.
(cited in pages 118, 120)

[99] E. Baştuğ, J.-L. Guénégo, and M. Debbah, "Proactive small cell networks," in *ICT 2013*, Casablanca, Morocco, May 2013.
(cited in page 118)

[100] V. A. Siris, X. Vasilakos, and G. C. Polyzos, "Efficient proactive caching for supporting seamless mobility," *arXiv preprint: 1404.4754*, 2014.
(cited in page 118)

[101] M. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *IEEE Transactions on Information Theory*, vol. 60, no. 5, pp. 2856–2867, May 2014.
(cited in page 118)

[102] J. Hachem, N. Karamchandani, and S. Diggavi, "Coded caching for heterogeneous wireless networks with multi-level access," *arXiv preprint: 1404.6560*, 2014.
(cited in page 118)

[103] N. Golrezaei, K. Shanmugam, A. G. Dimakis, A. F. Molisch, and G. Caire, "Femtocaching: Wireless video content delivery through distributed caching helpers," in *IEEE INFOCOM*. IEEE, 2012, pp. 1107–1115.
(cited in page 118)

[104] M. Ji, G. Caire, and A. F. Molisch, "Wireless device-to-device caching networks: Basic principles and system performance," *arXiv preprint: 1305.5216*, 2013.
(cited in page 118)

[105] J. Pääkkönen, C. Hollanti, and O. Tirkkonen, "Device-to-device data storage for mobile cellular systems," in *IEEE Globecom Workshops (GC Wrokshops)*, December 2013, pp. 671–676.
(cited in page 118)

[106] K. Hamidouche, W. Saad, and M. Debbah, "Many-to-many matching games for proactive social-caching in wireless small cell networks," in *WNC3 workshop in conjuction with WiOpt*, Hammamet, Tunisia, May 2014.
(cited in page 118)

[107] F. Pantisano, M. Bennis, W. Saad, and M. Debbah, "Match to cache: Optimizing user association and backhaul allocation in cache-aware small cell networks," *[Online] http://goo.gl/OgMGLo*, 2015.
(cited in page 118)

[108] P. Blasco and D. Gunduz, "Learning-based optimization of cache content in a small cell base station," *arXiv preprint: 1402.3247*, 2014.
(cited in page 118)

[109] J. N. Tsitsiklis, D. Bertsekas, and M. Athans, "Distributed asynchronous deterministic and stochastic gradient optimization algorithms," *IEEE Transactions on Automatic Control*, vol. 31, no. 9, pp. 803–812, 1986.
(cited in page 118)

[110] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and zipf-like distributions: Evidence and implications," in *IEEE INFOCOM'99*, vol. 1. IEEE, 1999, pp. 126–134.
(cited in page 119)

[111] J. Qin, Y. Chow, J. Yang, and R. Rajagopal, "Distributed online modified greedy algorithm for networked storage operation under uncertainty," 2014.
(cited in page 136)