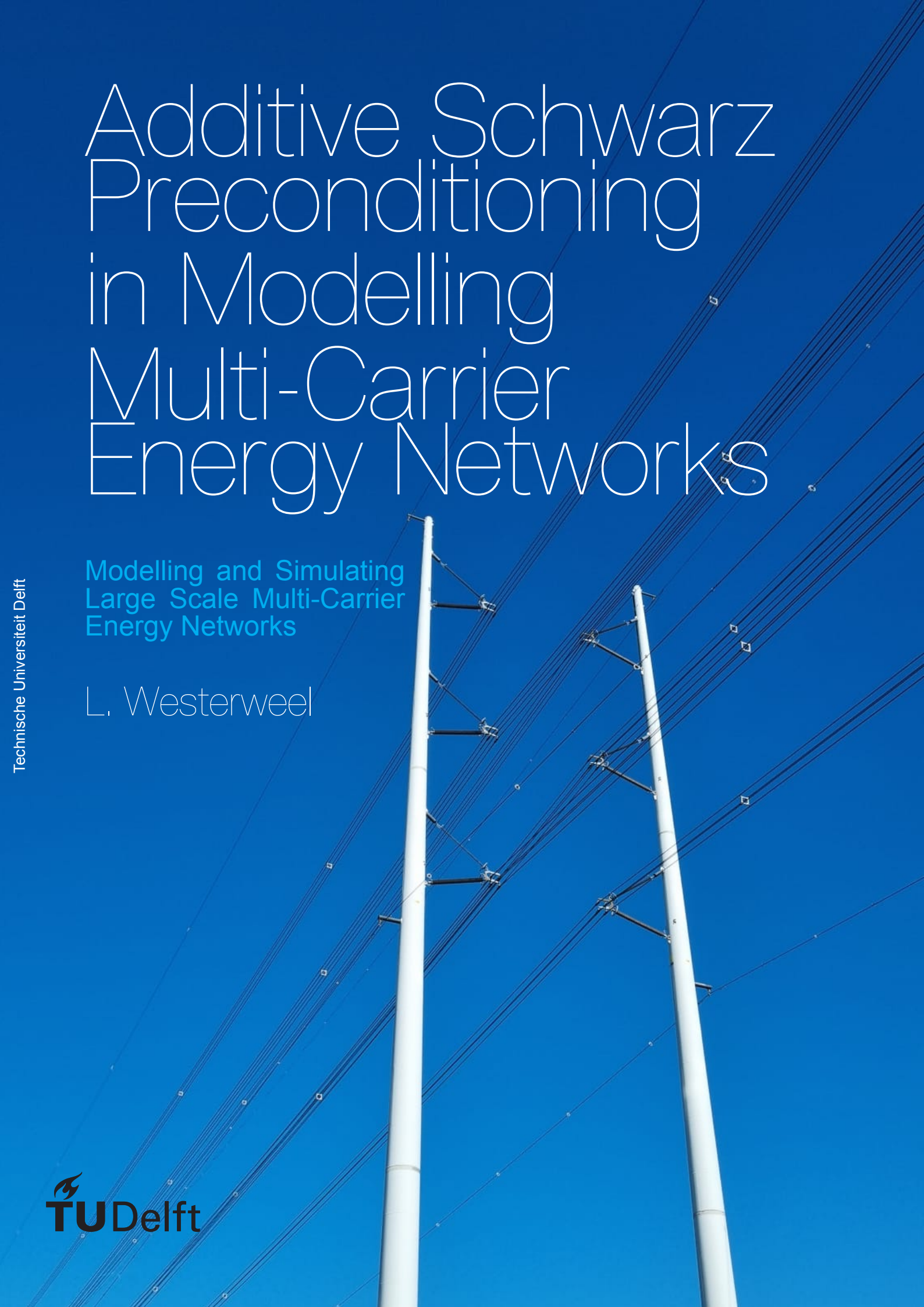
Additive Schwarz Preconditioning in Modelling Multi-Carrier Energy Networks

Modelling and Simulating Large Scale Multi-Carrier Energy Networks

L. Westerweel

Technische Universiteit Delft



TU Delft

Additive Schwarz Preconditioning in Modelling Multi-Carrier Energy Networks

Modelling and Simulating Large Scale Multi-Carrier Energy Networks

by

L. Westerweel

to obtain the degree of Master of Science in Applied Mathematics

at the Delft University of Technology,

to be defended publicly on June 24 2026, at 14:00.

Student number:	4792548	
Project duration:	August 15, 2025 – June 24, 2026	
Thesis committee:	Prof. dr. ir. C. Vuik,	TU Delft, supervisor
	B. Nguyen,	TU Delft, daily supervisor
	Dr. ir. H. X. Lin,	TU Delft, external supervisor

An electronic version of this thesis is available at http://repository.tudelft.nl/.

Cover image: Photo by J. Wouters.

TU Delft

Preface

This thesis marks the final step in completing my Masters' degree in Applied Mathematics at TU Delft and my time as a student. Since 2018, I have been working hard on obtaining both a Bachelors' and Masters' degree in Applied Mathematics, which has been both challenging and fun. I have also participated in plenty of extra-curricular activities, therefore I would like to thank the TU Delft and W.I.S.V. 'Christiaan Huygens' for all the opportunities I had, allowing me to learn a lot about myself and develop both professionally and personally.

During the last 10 months, I really enjoyed working on this research and report. I would like to thank my supervisors, Kees Vuik and Buu-Van Nguyen, for their engaging supervision and guidance, and experienced input. Buu-Van, I am especially thankful for all your time spent reviewing this report, debugging code together and answering all my questions during our weekly meetings. I would also like to thank Johan Romate for joining the intermediate presentations of this research and your valuable input. Thank you Hai Xiang Lin for joining my thesis committee as an external supervisor and taking the time to review this research.

Finally, I am very grateful for my friends, family and partner Jaron, for their support, interest and patience not only during this thesis project, but during my entire studies.

L. Westerweel
Delft, June 2026

Abstract

The incorporation of renewable energy sources into daily life is necessary to reduce the emission of greenhouse gases and combat climate change. Due to the variability of these renewable sources and the mixing of different energy carriers, this is not a straightforward task. It is important to have reliable and fast simulations that can show the effects of coupling various energy carriers, to be used by policy makers and network operators. Direct solution methods are often used, but these scale poorly in problem size, increasing computational efforts significantly. It is therefore recommended to use preconditioned iterative solvers for large energy network simulations.

This research investigates the performance of the additive Schwarz (AS) preconditioner in the solution process of the steady-state load flow equations of large single-carrier and coupled energy networks. This is done by defining the construction of the AS preconditioner for gas, electricity and coupled networks, and by presenting the results of its application to both small- and large-scale networks. Various AS-related parameters are varied to study its effect on the accuracy, speed, robustness and scalability of the load flow solution process.

This study shows that the AS preconditioner can successfully accelerate convergence of the steady-state load flow equations of single-carrier and coupled energy networks, resulting in an algorithmically stable solution process that maintains feasibility for large energy networks. The restricted AS preconditioner variant is preferred in terms of accuracy, speed and robustness. For large-scale single-carrier energy networks, reusing the AS preconditioner of a previous non-linear iteration can save in computational costs, while not significantly affecting the final solution. Additionally, this research found several aspects of the used gas network models that can hinder or negatively impact the effectiveness of the AS preconditioner. Solutions and alternatives are also presented. Future work can consist of further developing and investigating the approach of constructing AS preconditioners, by considering different partitioning strategies, parallel implementations, two-level variants and larger test datasets.

Contents

List of Abbreviations

AC alternating current.
AS additive Schwarz.

Bi-CG bi-conjugate gradient.
Bi-CGSTAB bi-conjugate gradient stabilized.
BIMs basic iterative methods.

CG conjugate gradient.
CHP combined heat and power.
CPU central processing unit.

DD domain decomposition.
DHS district heating system.

FDLF fast decoupled load flow.

GFG gas-fired generator.
GHV gross heating value.
GMRES generalized minimum residual.
GPU graphics processing unit.

ILU incomplete LU.

KCL Kirchhoff's current law.
KVL Kirchhoff's voltage law.

MCN multi-carrier energy network.
MES multi-energy system.
MS multiplicative Schwarz.
MSS master-slave splitting.

NR Newton-Raphson.

P2G power-to-gas.

RAS restricted additive Schwarz.
RMS root mean squared.

SAP Schwarz alternating procedure.
SC single-carrier.
SPD symmetric positive definite.

1 Introduction

Climate change is one of the biggest challenges of the twenty-first century [16]. Since the industrial revolution, human actions have increased the atmospheric concentration of greenhouse gas emissions [17]. The shifts in recent global climate patterns, primarily driven by human activities, are referred to as *contemporary climate change*. It involves shifts in long-term averages and variations, with extreme weather events occurring more frequently [17, 25]. By 2017, the average temperature had increased with $1.0\,^{\circ}\mathrm{C}$ compared to pre-industrial levels [25]. To combat climate change, the Paris Agreement [59] was signed and adopted by 195 parties at the UN Climate Change Conference (COP21) in Paris in 2015. Its overarching goal is to

> *"Hold the increase in the global average temperature to well below* $2.0\,^{\circ}\mathrm{C}$ *above pre-industrial levels and to pursue efforts to limit the temperature increase to* $1.5\,^{\circ}\mathrm{C}$ *above pre-industrial levels."*
>
> —Paris Agreement, 2015

Additionally, the Dutch Climate Act that was signed in 2019 [32] states that

> *"The Netherlands must be climate neutral in 2050 with an energy system with greenhouse gas emissions that are* 95% *lower than 1990."*
>
> —Dutch Climate Act, 2019

To achieve climate goals such as the Paris Agreement and Dutch Climate Act, the emissions of greenhouse gases have to be reduced. This requires fossil fuel burning to be significantly reduced, while rapidly growing the share of sustainable energy. This type of energy largely consists of *renewable energy*, which is derived from natural sources that are replenished on a human timescale, such as wind or solar power. It is projected that ultimately renewable energy will account for the vast majority of electricity generation around 2050 [52]. These renewable energy sources are often inherently intermittent and variable, meaning that they are not constantly available [14, 55]. Incorporation of renewables is therefore not straightforward. Energy networks that include more than one carrier of energy, such as electricity, heat or hydrogen gas, are referred to as multi-carrier energy networks (MCNs) [35]. In order to properly incorporate renewable energy in MCNs, it is important to have reliable and fast simulations that can show the effects of coupling various energy carriers [5, 52], to be used by policy makers or network operators [46].

1.1. Load Flow Analysis of Energy Systems

Energy systems can often be represented by a graph, consisting of links and nodes, which visualise the flow of energy through the network via the links, and the consumption or generation of energy at certain nodes. Electricity and gas networks are based on different physical laws (see Chapter 2), but share this energy network graph structure. Analysis of these energy systems is often done by determining the flow of energy within the network, also referred to as 'load-flow analysis'. Mathematically, the load-flow analysis leads to the network's load-flow equations (see Chapter 3). The solution of these

load-flow equations results in the flow of energy within the network, when a steady-state is reached. The steady-state implies that the system has settled, such that there is no more change happening over time.

The load-flow equations are often non-linear and lead to large-scale systems. Extensive research has been conducted into solving the steady-state load flow equations of electricity networks, varying from small- to large-scale applications [1, 10, 12, 23, 24, 34, 38, 43, 47, 48]. Analysis of solution methods for the steady-state load flow of gas networks has also been conducted, but not as extensively as for electricity networks [40, 41, 45]. The coupling of energy networks into MCNs introduces coupling units and unifies systems operating with potentially different physical laws, increasing model complexity (see Chapter 4). In recent studies, it has been shown that a steady-state load-flow analysis can be performed for MCNs [26, 35, 37, 62, 64, 65].

The most commonly used and standard approach to solve the non-linear steady-state load flow equations of energy systems is Newton-Raphson (NR) [10, 20, 23]. This is an iterative method where the system is linearised around the current network state. Solving the linearised system involves the Jacobian matrix and results in an updated state of the network. This process is repeated until the load flow equations are sufficiently satisfied. Various linear solution methods are used to solve the linearised systems, such as a direct solve, basic iterative methods, or Krylov-type methods (see Chapter 5). Computing the solution of the linearised systems can be expensive and tedious, especially when the corresponding energy network is large and complex. Extensive research has been done on linear solution methods, considering different applications [49]. The efficiency of the overall load flow calculation is strongly influenced by the efficiency of the linear solver and the Jacobian matrix [48].

Direct solvers scale poorly in problem size, increasing the computational efforts needed to converge significantly [23]. It is generally recommended to use iterative solvers for solving large power flow problems [12, 38, 34]. The same results have been observed for gas networks [45, 40]. However, for smaller networks, the direct solver shows an advantage over the iterative (preconditioned Krylov) solvers due to their big overhead [45]. Thus, for smaller energy networks a direct solver is more suitable, while for larger networks the class of Krylov solvers should be considered [63].

Preconditioning is a technique used to improve the efficiency and robustness of iterative solution methods, such as Krylov solvers (see Chapter 6) [49]. A class of preconditioners that is often used in energy systems modelling is based on an incomplete LU (ILU) factorisation. Various studies investigated the application of ILU preconditioners in the linear solution process of energy networks [23, 48], generally showing good performance in terms of accuracy, speed and scalability. The application of ILU preconditioners to gas networks has not been studied as extensively. It was even concluded that the ILU preconditioner is not suitable for load flow analysis of gas networks [40].

A different class of preconditioners is based on the principle of divide-and-conquer. This principle refers to methods where the global domain is split into several subdomains, so that solving the local problems sequentially leads to solving the global problem. This class of methods is referred to as domain decomposition (DD). Such DD methods have been applied in the load flow solution process of electricity, gas and district heating networks, although test problems were mostly small- or medium-scale [34, 56, 64]. A commonly used class of DD preconditioners is based on Schwarz alternating procedures (SAPs), such as the multiplicative Schwarz (MS) and additive Schwarz (AS) preconditioners [49]. The performance of AS variants was studied in a power flow setting [1, 2, 19], but not yet for gas and coupled networks. In the context of energy network modelling, a partition of the global network graph can either be done based on the physical properties of the network, or on the mathematical properties of the network graph. Research on partitioning methods has been done in other fields where DD is often used and in some single-carrier (SC) energy network settings, but not in the context of gas and coupled energy networks.

1.2. Scope and Boundaries of the Research

From the literature review, several conclusions and recommendations can be drawn. Integrated solution methods for MCNs are at an early stage and present research opportunities. The numerical performance of solution methods is the limiting factor for fast, robust and accurate simulations for large scale MCNs and required research into preconditioned iterative methods using a decomposition of the network. The selection of an appropriate preconditioner can be as important as the choice of iterative solution method. Furthermore, the performance of a preconditioned iterative method is problem-dependent and the number of hyperparameters often makes them highly customizable. The application of preconditioners based on SAPs and graph-based partitioning algorithms applied to gas and coupled networks present research opportunities. Additionally, DD methods applied to SC energy networks have shown promising results thus far [1, 34, 56]. Based on these literature findings and recommendations, the scope of this research consists of the application of AS (an SAP preconditioning strategy) in electricity, gas and coupled energy networks.

The application of ILU preconditioners in power flow problems has been studied extensively and is generally found to perform well. ILU preconditioning is not appropriate for gas network simulations, and alternative preconditioning strategies are therefore an open problem. The ILU preconditioner will therefore not be further investigated in this research. Additionally, dynamic energy simulations, market-based models, control strategies, and optimization problems are considered outside the scope for this research, and should be considered as applications of the models mentioned in this report. Furthermore, the formulation and models used to represent energy networks are mentioned and discussed in this thesis report, but are not part of the scope of the research.

1.3. Research Questions

Based on the presented literature and the scope of this research, the preliminary research question is formulated as follows:

How does an additive Schwarz preconditioner perform when applied to a Krylov solver in the solution process of the steady-state load flow equations of large scale coupled gas-electricity energy networks?

In order to answer this research question, there are several sub-questions formulated to support the main research goal:

1. How does an additive Schwarz preconditioner perform when solving steady-state load flow equations of gas and electricity networks separately?
2. How do different partitioning strategies affect the performance of the additive Schwarz preconditioner when solving steady-state load flow equations of gas, electricity and coupled networks?
3. How do the basic, interpolated and restricted variants of the additive Schwarz preconditioner compare when solving steady-state load flow equations of gas, electricity and coupled networks?
4. What is the effect of coupling gas and electricity networks on the performance of the additive Schwarz preconditioner when solving steady-state load flow equations of coupled networks?

1.4. Thesis Structure

The rest of this thesis is structured as follows:

- Chapter 2 ‘Single-Carrier Energy Systems’ provides an introduction to the relevant topics related to gas and electrical energy systems, required to derive the steady-state load flow equations of gas and electrical networks.
- Chapter 3 ‘Modelling Single-Carrier Energy Networks’ contains the derivation and discussion of the steady-state load flow equations of SC gas and electrical networks, based on the physical concepts discussed in Chapter 2.
- Chapter 4 ‘Modelling Multi-Carrier Energy Networks’ discusses the modelling of multi-carrier energy networks. The methods used for coupled energy systems and the coupled load flow equa-

tions are derived and discussed.

- Chapter 5 ‘Linear and Non-Linear Solution Methods’ gives an overview of the linear and non-linear solution methods used in literature and this research, focussing on linear Krylov methods and the non-linear Newton Raphson method. Additionally, the usage of linear and non-linear solution methods in recent studies on energy systems are discussed.
- Chapter 6 ‘Preconditioning’ introduces the principle of preconditioning a linear system. The preconditioners related to incomplete LU factorization and alternating Schwarz procedures are discussed in more detail. The usage of the before-mentioned preconditioners in recent studies on energy systems is also presented.
- Chapter 7 ‘Partitioning of Energy Networks’ gives an overview of multi-level graph partitioning algorithms often applied in the solution process of energy networks.
- Chapter 8 ‘Methodology and Experimental Setup’ discusses the setup of the conducted experiments as well as the methodology used to construct and analyse the results to answer the presented research questions.
- Chapter 9 ‘Additive Schwarz Preconditioning for Single-Carrier Energy Networks’ presents the results of the application of the AS preconditioner in the solution process of SC energy networks. First, the results for small-scale gas networks are analysed in detail, after which the same is done for small-scale electricity networks. The chapter ends with a section presenting the results of large-scale gas and electricity networks.
- Chapter 10 ‘Additive Schwarz Preconditioning for Coupled Multi-Carrier Energy Networks’ presents the results of the application of the AS preconditioner in coupled energy networks, focussing on the effect of coupling and interaction between the different carriers.
- Chapter 11 ‘Conclusion and Discussion’ summarizes the conclusions drawn from the conducted experiments and provides answers to the research questions. Additionally, the limitations of this research and possibilities for future research are discussed.

2

Single-Carrier Energy Systems

The modelling of large scale multi-carrier energy networks (MCNs) is based on the coupling and interaction between different energy carriers, such as electricity, gas and heat. These carriers are based on different physical concepts and operate on different time-scales. In order to interpret and work with the models and equations of MCNs, it is important to have a good understanding of the physics and underlying principles of each energy carrier individually. The aim of this chapter is to get a basic understanding of the physical phenomena in each single-carrier (SC) network considered in this research, focussing on mathematical formulation of energy network components. The concepts of this chapter are used in Chapter 3 to formulate mathematical models of SC energy networks. The contents and aim of this chapter are based on the work of Markensteijn [35].

This chapter gives the concepts and models used in the modelling of electricity and gas networks. Electricity networks are discussed in Section 2.1, by introducing fundamental concepts and transmission line modelling. Section 2.2 introduces gas networks by considering gas mass flow through a pipe and other gas network elements.

2.1. Electricity Networks

Electricity plays a central role in many modern technologies as a carrier of electric energy. It is based on *electric charge*, a physical property of matter that can result in attraction or repulsion in the presence of other matter with charge. The movement of this electric charge is known as *electric current*. The charge can flow along a closed path, referred to as a *circuit*, and is used to power components, such as lamps, motors and other electronic devices. In practice, electricity is generated at a centralized source and transported over power lines to the end users.

This section provides an overview of electricity as an energy carrier and introduces some fundamental concepts. Additionally, the modelling of power transmission lines is presented. The contents of this section are based on [51, 6, 3, 31].

2.1.1. Electricity Network Fundamentals

Fundamental concepts of electricity networks include voltage, current and power, as well as important physical laws and properties of circuit elements.

Voltage and Current in AC Circuits

A circuit with alternating current (AC) is one where the current alternates between positive and negative over time. The currents and voltages in an AC power system usually follow a time-dependent sinusoidal function. In a steady-state system, the behaviour of the system no longer varies over time. In an AC circuit this implies that the current and voltage still follow a time-dependent sinusoid, but with a fixed frequency ω. The steady-state voltage is then as follows

$$v(t) = V_m \cos(\omega t + \delta_V),$$

with V_m the voltage amplitude in volt (V), ω the angular frequency in hertz (Hz) and δ_V the phase of the voltage. Similarly, for the current we have

$$i(t) = I_m \cos(\omega t + \delta_I),$$

with I_m the current amplitude in ampere (A), ω the angular frequency in hertz (Hz, and δ_I the phase of the current. The fixed angular frequency ω is usually 50 or 60 Hz for domestic use, depending on the country or region.

In AC power system modelling it is customary to work with the *root mean squared (RMS) value*, denoted with $|\cdot|$, which is the value of the equivalent direct voltage (or current) that dissipates the same amount of power in a given resistor during one time period of the alternating voltage (or current). They are computed as

$$|V| = \sqrt{\frac{1}{T}\int_0^T v(t)^2 \, dt} = \frac{1}{\sqrt{2}} V_m,$$
$$|I| = \sqrt{\frac{1}{T}\int_0^T i(t)^2 \, dt} = \frac{1}{\sqrt{2}} I_m.$$

Without loss of generality, we can write the sinusoidal expressions for voltage and current as

$$v(t) = \sqrt{2}|V| \cos(\omega t), \tag{2.1}$$
$$i(t) = \sqrt{2}|I| \cos(\omega t - \phi). \tag{2.2}$$

Here, *phase shift angle* ϕ represents the difference in phase of the voltage and current sinusoids.

Phasor Notation

A *phasor* is a complex number that represents the amplitude and phase of a sinusoid. The angular frequency ω is constant in a steady-state system. Therefore, we are only interested in the amplitude and phase shift of the voltages and currents, which is why phasor notation is widely used. In the phasor domain the time-dependency is not denoted, as the frequency is constant in a steady-state system. The use of phasor notation allows to switch from the time domain to the phasor domain, so that time-dependency is dropped, simplifying steady-state calculations and analysis. Using Euler's identity, we see the equivalence relation between a sinusoid, complex number and phasor as follows

$$v(t) = \sqrt{2}|V| \cos(\omega t + \delta_V) = \sqrt{2}\mathrm{Re}\left(|V| e^{\mathbf{i}(\omega t + \delta_V)}\right) = \sqrt{2}\mathrm{Re}\left(V e^{\mathbf{i}\omega t}\right),$$

where $V = |V| e^{\mathbf{i}\delta_V}$ is the voltage phasor. In phasor notation we denote the amplitude and phase angle,

$$V = |V| e^{\mathbf{i}\delta_V} = |V| \angle \delta_V.$$

Resistance and Ohm's Law

Various materials are known to resist the flow of electric charge, where the amount of resistance to the flow is a material property. This *resistance* is measured in ohms (Ω). *Ohm's law* states that the voltage v across an element (resistor) is directly proportional to the current i flowing through. This proportionality is defined as the resistance R of an element, which in computed in the phasor domain as

$$R = \frac{V}{I}. \tag{2.3}$$

Kirchhoff's Voltage and Current Laws in AC Systems

Besides Ohm's law, there are two other important physical laws often used in circuit analysis. There are *Kirchhoff's current law (KCL)* and *Kirchhoff's voltage law (KVL)*.

- KCL is based on conservation of charge and states that the sum of current phasors entering and leaving a node is always zero,

$$\sum_{n=1}^{N} I_n = 0. \tag{2.4}$$

  This implies that the magnitude of currents entering the node equals the magnitude of currents leaving the node.

- KVL is based on conservation of energy and states that the sum of all voltage phasors around a closed path is zero,

$$\sum_{m=1}^{M} V_m = 0.$$

  This implies that the sum of all voltage drops within a closed loop equals the sum of all voltage rises.

Power

The concept of power is defined as the time rate of the expending and absorbing energy, and is measured in watts ($\mathrm{W} = \mathrm{J\,s^{-1}}$). The *instantaneous power* $p(t)$, is the power absorbed or supplied by an element in a circuit at a specific time. It is computed as the product of the voltage across the element and the current through it,

$$p(t) = v(t) \cdot i(t). \tag{2.5}$$

Substitution of voltage and current definitions (2.1) and (2.2) into instantaneous power (2.5) results in

$$\begin{aligned} p(t) &= 2|V||I| \cos(\omega t) \cos(\omega t - \phi), \\ &= |V||I| \cos(\phi)\,(1 + \cos(2\omega t)) + |V||I| \sin(\phi) \sin(2\omega t). \end{aligned} \tag{2.6}$$

In an AC circuit, the voltage and current are constantly changing over time, and thus the power is as well. Similarly to how RMS values of voltage and current are used in AC circuits, the *real power* P is defined as the average of the instantaneous power during one time period,

$$P = \frac{1}{T} \int_0^T p(t)\, dt = |V||I| \cos(\phi). \tag{2.7}$$

Note that the real power P corresponds to the time-independent part of the first term in (2.6). The real power can be interpreted as the useful power that is actually dissipated by the load, and is measured in watt (W).

Besides the real power, in AC systems there is also a *reactive power*, denoted by Q. This is the power that oscillates back and forth between the source and the load, and does not do work. It is however a necessary part for the functioning of AC systems. The reactive power is measured in volt-ampere reactive (VAR) and corresponds to the time-independent part of the second term in (2.6), which has an average of zero. We thus have

$$Q = |V||I| \sin(\phi). \tag{2.8}$$

Both the real and reactive power depend on phase shift angle ϕ, which indicates how in phase the voltage and current are. If $\phi > 0$, the current is lagging behind the voltage, and when $\phi < 0$, the current is leading the voltage. The *power factor* is defined as

$$\mathrm{pf} = \cos(\phi) = \cos(\delta_V - \delta_I).$$

The phase shift angle ϕ influences the magnitude of the real and reactive powers. If ϕ is zero, the power factor is 1 and the voltage and current sinusoids are in phase, resulting in a large true power P, and zero reactive power Q. This can be interpreted as a perfect system where all electricity from the source is used to do work. On the other hand, when ϕ is near $\pm\frac{\pi}{2}$, and the power factor is thus near 0, the true power P becomes zero, and reactive power Q becomes large. This implies that no electricity

is used to do work and is instead stored in magnetic or electric fields.

In order to simplify power relations and expressions, the *complex power* S is used. It combines the real and reactive powers into a single complex number, defined as

$$S = VI^*, \tag{2.9}$$

the product of the voltage phasor V and complex conjugate of current phasor I. The complex power can equivalently be written in polar coordinates, phasor notation and Cartesian coordinates as follows

$$\begin{aligned} S = VI^* &= |V||I|\left(\cos(\phi) + \mathbf{i}\sin(\phi)\right), \\ &= |V||I|e^{\mathbf{i}\phi}, && \text{Polar} \\ &= |V||I|\angle\phi = |S|\angle\phi, && \text{Phasor} \\ &= P + \mathbf{i}Q. && \text{Cartesian} \end{aligned}$$

In the phasor notation, the complex power has magnitude $|S| = |V||I| = \sqrt{P^2 + Q^2}$, defined as the *apparent power*. This is a practical quantity often used as an electrical apparatus specification.

Impedance and Admittance
Besides resistors, there are also other passive circuit elements. The most relevant in power system analysis are the storage elements *capacitors* and *inductors*. A resistor dissipates energy, whereas capacitors and inductors store energy.

A capacitor stores energy in an electric field. The amount of energy stored is proportional to the applied voltage. The effect of a capacitor in an AC circuit is that the voltage sinusoid will lag behind the current sinusoid, making them out of phase. With a lagging voltage there is a decrease in the real power and increase in the reactive power. The energy stored in the electric field is indicated as the reactive power.

Similarly, an inductor stores energy, but in a magnetic field. The amount of energy stored is proportional to the current flowing through. The effect of an inductor in an AC circuit is that the current sinusoid will lag behind the voltage sinusoid, making them out of phase as well. With a lagging current there is a decrease in the real power and increase in the reactive power. The energy stored in the magnetic field is indicated as the reactive power.

Similar to the complex power notation, the resistance, capacitance and inductance as voltage-current relations in the phasor domain, can be combined into one complex quantity, the *impedance* Z. Impedance represents the opposition that the circuit exhibits to the flow of sinusoidal current. The general expression of the impedance is given as

$$Z = \frac{V}{I} = R + \mathbf{i}X, \tag{2.10}$$

a complex number measured in ohm (Ω). The real component R is the resistance and the imaginary component X is the *reactance*. If the reactance is positive, so $X > 0$, the component is inductive and results in a lagging current. If the reactance is negative, $X < 0$, the component is capacitive and results in a lagging voltage. Note that impedance is a ratio of two phasors, but is not itself a phasor.

Conversely, *admittance* Y represents how easily current flows through a circuit. It is defined as the reciprocal of the impedance,

$$Y = \frac{1}{Z} = \frac{R}{R^2 + X^2} + \mathbf{i}\frac{X}{R^2 + X^2} = G + \mathbf{i}B. \tag{2.11}$$

The real component G is referred to as the *conductance*, while the imaginary component B is called the *susceptance*. Similar but opposite to the reactance, if $B > 0$ the component is capacitive, while it is inductive when $B < 0$.

2.1.2. Transmission Line Modelling

Transmission lines are an important part of power systems modelling, as they transport electricity to the end user. The derivation of different transmission line models can be found in more detail in Chapter 4 of [6] and Appendix E of [51]. This section discusses the properties of transmission lines and uses the derivation of the transmission and admittance matrices to relate sending currents, receiving currents and related voltages. Besides transmission lines, there are also other electricity network components that have to be modelled, most commonly transformers. More about the modelling of transformers can be found in Chapter 6 of [31], Chapter 3 of [51] and Chapter 13 of [3].

There are four different important parameters to consider for the modelling of a transmission line, which are *series resistance*, *series inductance*, *shunt capacitance* and *shunt conductance*. The series resistance and inductance refer to resistance and voltage lagging in line with the path of the current, whereas shunt capacitance and conductance refer to leakage current that is in parallel with the transmission line. The series impedance can be written as

$$z = r + \mathbf{i}\omega x,$$

where z is the series impedance, r the series resistance and x the series inductance. Similarly, the shunt admittance is given by

$$y = g + \mathbf{i}\omega b,$$

where y the shunt admittance, g the shunt conductance and b the shunt capacitance.

An equivalent circuit of a transmission line can be found in Figure 2.1. The left side of the circuit, with subscripts i is the sending side, while the right side corresponds to the receiving side, indicated with subscripts j. The shunt admittance is split into half, with one part on the sending end and one part on the receiving end of the line. Define the *total series impedance* and *total shunt capacitance* per phase as

$$Z = zL, \quad Y = yL,$$

where L is the transmission line length in meter (m). The equivalent circuit given in Figure 2.1 is a so-called *lumped equivalent circuit*. The term 'lumped' refers to the assumption that circuit components are concentrated at discrete points, rather than being spread out along the entire line. Therefore, we define the equivalent lumped series impedance and shunt admittance as

$$Z' = Z \cdot \frac{\sinh(\gamma L)}{\gamma L}, \quad Y' = Y \cdot \frac{\tanh(\gamma L/2)}{\gamma L/2},$$

respectively. Here, $\gamma = \sqrt{zy}$ is the *propagation constant*. Notice the dependence of the lumped impedance and admittance on the transmission line length L.

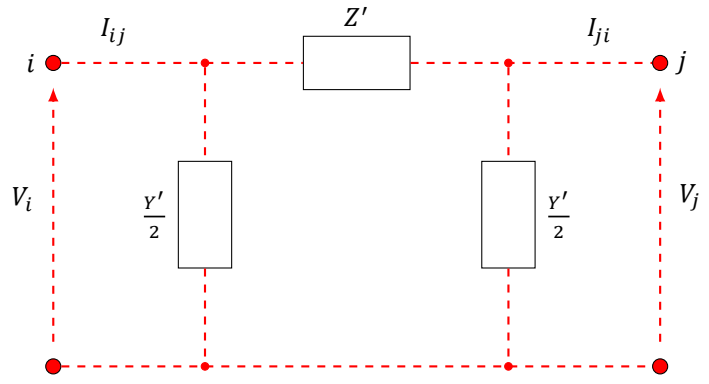


Figure 2.1: Equivalent lumped circuit of a long transmission line.

Transmission parameters refer to the parameters that relate the input variables V_i, I_{ij} to the output variables V_j, $-I_{ji}$. The corresponding *transmission matrix* of the lumped equivalent circuit in Figure 2.1

follows from two-port theory and is given as

$$\begin{bmatrix} V_i \\ I_{ij} \end{bmatrix} = \begin{bmatrix} \frac{Z'Y'}{2} + 1 & Z' \\ \left(\frac{Z'Y'}{4} + 1\right) Y' & \frac{Z'Y'}{2} + 1 \end{bmatrix} \begin{bmatrix} V_j \\ -I_{ji} \end{bmatrix}. \tag{2.12}$$

Using this transmission matrix for the lumped equivalent circuit we can express sending current I_{ij} in terms of the sending and receiving voltages V_j, V_i,

$$I_{ij} = \left(V_i - V_j\right) \frac{1}{Z'} + \frac{1}{2} Y' V_i. \tag{2.13}$$

Note that impedance and admittance are related through (2.11). Therefore, we will from now on denote the series elements with the superscript $\cdot^{\text{se}}$, and the shunt elements with the superscript $\cdot^{\text{sh}}$. Substituting Z' and Y' and working out the series and shunt admittances, (2.13) becomes

$$\begin{aligned} I_{ij} &= \left[Y'^{\text{se}} + \frac{1}{2} Y'^{\text{sh}} \right] V_i - Y'^{\text{sh}} V_j, \\ &= \left[\frac{1}{C_1} (G_{ij}^{\text{se}} + \mathbf{i} B_{ij}^{\text{se}}) + \frac{C_2}{2} (G_{ij}^{\text{sh}} + \mathbf{i} B_{ij}^{\text{sh}}) \right] V_i - \left[\frac{1}{C_1} (G_{ij}^{\text{se}} + \mathbf{i} B_{ij}^{\text{se}}) \right] V_j, \\ &= \Upsilon_{ii,j} V_i - \Upsilon_{ij} V_j, \end{aligned} \tag{2.14}$$

where $C_1 = \frac{\sinh(\gamma L)}{\gamma L}$, $C_2 = \frac{\tanh(\gamma L/2)}{\gamma L/2}$ and coefficients

$$\begin{aligned} \Upsilon_{ii,j} &= \frac{1}{C_1} (G_{ij}^{\text{se}} + \mathbf{i} B_{ij}^{\text{se}}) + \frac{C_2}{2} (G_{ij}^{\text{sh}} + \mathbf{i} B_{ij}^{\text{sh}}), \\ \Upsilon_{ij} &= \frac{1}{C_1} (G_{ij}^{\text{se}} + \mathbf{i} B_{ij}^{\text{se}}). \end{aligned}$$

These coefficients describe the relation between the sending current I_{ij} and related voltages V_i and V_j. The term $\Upsilon_{ii,j}$ is the sum of the shunt and series admittances connected to element i via the transmission line with sending current I_{ij}. The term Υ_{ij} is the series admittance of the direct transmission line between elements i and j. Since admittances are a property of the line itself, $\Upsilon_{ij} = \Upsilon_{ji}$ and $\Upsilon_{ii,j} = \Upsilon_{jj,i}$. The relations from the transmission matrix (2.13) can also be represented by the matrix

$$\begin{bmatrix} I_{ij} \\ -I_{ji} \end{bmatrix} = \begin{bmatrix} \Upsilon_{ii,j} & -\Upsilon_{ij} \\ -\Upsilon_{ij} & \Upsilon_{ii,j} \end{bmatrix} \begin{bmatrix} V_i \\ V_j \end{bmatrix}.$$

The *total current* at element i is defined as the sum of all sending currents leaving the element,

$$I_i = -\sum_{j, j \neq i} I_{ij} = -\sum_{j, j \neq i} \left[\Upsilon_{ii,j} V_i - \Upsilon_{ij} V_j \right]. \tag{2.15}$$

Substitution of the sending current expression in terms of voltages (2.14) into the total current leads to the *admittance matrix* $\mathbf{Y}$,

$$\begin{aligned} \begin{bmatrix} I_1 \\ I_2 \\ \vdots \\ I_N \end{bmatrix} &= -\begin{bmatrix} \sum_{j, j \neq 1} \Upsilon_{11,j} & -\Upsilon_{12} & \cdots & -\Upsilon_{1N} \\ -\Upsilon_{12} & \sum_{j, j \neq 2} \Upsilon_{22,j} & \cdots & -\Upsilon_{2N} \\ \vdots & \cdots & \cdots & \vdots \\ -\Upsilon_{1N} & -\Upsilon_{2N} & \cdots & \sum_{j, j \neq N} \Upsilon_{NN,j} \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ \vdots \\ V_N \end{bmatrix}, \\ &= -\begin{bmatrix} Y_{11} & Y_{12} & \cdots & Y_{1N} \\ Y_{21} & Y_{22} & \cdots & Y_{2N} \\ \vdots & \cdots & \cdots & \vdots \\ Y_{N1} & Y_{N2} & \cdots & Y_{NN} \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ \vdots \\ V_N \end{bmatrix}, \end{aligned} \tag{2.16}$$

with coefficients

$$Y_{ii} = \sum_{j, j \neq i} \Upsilon_{ii,j}, \; Y_{ij} = Y_{ji} = -\Upsilon_{ij}.$$

The admittance matrix of a circuit can be constructed by observing that the elements are determined as:

- The diagonal terms Y_{ii} are the sum of all admittances directly connected to element i.
- Y_{ij} equals the negative value of the admittance connected between elements i and j.

With the derived expression of sending current in terms of voltages (2.14), we can formulate the complex power S_{ij} at the sending end of the transmission line using (2.9)

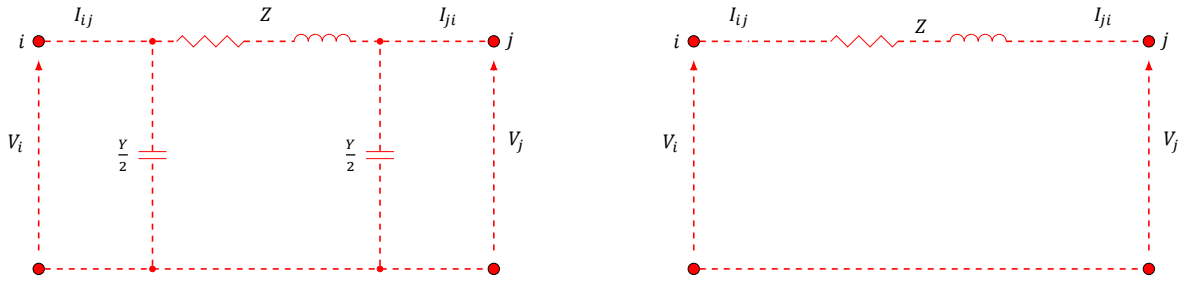
$$\begin{aligned} S_{ij} &= V_i I_{ij}^*, \\ &= |V_i| e^{\delta_{V_i}\mathbf{i}} \left[\left(|V_i| e^{-\delta_{V_i}\mathbf{i}} - |V_j| e^{-\delta_{V_j}\mathbf{i}} \right) \frac{G^{\text{se}} - \mathbf{i}B^{\text{se}}}{C_1} + \frac{1}{2} \left(G^{\text{sh}} - \mathbf{i}B^{\text{sh}} \right) C_2 |V_i| e^{-\delta_{V_i}\mathbf{i}} \right], \\ &= P_{ij} + \mathbf{i}Q_{ij}, \end{aligned} \tag{2.17}$$

with sending real and reactive powers

$$P_{ij} = \frac{|V_i|^2}{A} G^{\text{se}} - \frac{|V_i||V_j|}{C_1} \left[G^{\text{se}} \cos(\phi_{ij}) + B^{\text{se}} \sin(\phi_{ij}) \right] + \frac{1}{2} |V_i|^2 G^{\text{sh}} C_2, \tag{2.18}$$

$$Q_{ij} = -\frac{|V_i|^2}{A} B^{\text{se}} - \frac{|V_i||V_j|}{C_1} \left[G^{\text{se}} \sin(\phi_{ij}) - B^{\text{se}} \cos(\phi_{ij}) \right] - \frac{1}{2} |V_i|^2 B^{\text{sh}} C_2. \tag{2.19}$$

Note that the length of the transmission line L appears in the definition of the equivalent lumped impedance and admittance in constants C_1 and C_2. When modelling a transmission line, one therefore has to distinguish between long, medium and short length lines. In a long transmission line ($L > 240$ km) all components in the series impedance and shunt admittance are non-zero and thus have to be taken into account. For a medium length transmission line ($80 < L < 240$ km), simplifications can be made by assuming that $\sinh(\gamma L) \approx \gamma L$. Therefore, $Z' \approx Z$ and $Y' \approx Y$ ($C_1, C_2 = 1$). Additionally, we can assume that the shunt conductance g^{sh} becomes negligible. For a short transmission line ($L < 80$ km), even more simplifications apply such that the shunt capacitance b^{sh} also becomes negligible. For a short transmission line the shunt admittance Y is therefore assumed to be zero. The equivalent circuits of medium- and short-length transmission lines can be found in Figure 2.2.



(a) Medium transmission line. (b) Short transmission line.

Figure 2.2: Equivalent circuits of medium and short transmission lines, taken from [51].

2.2. Gas Networks

The energy carrier gas refers to any gaseous substance that is transported or consumed for its energy content. Some common gas energy carriers are natural gas, hydrogen and biogas. The gas is transported and converted into other energy carriers, such as electricity and heat, via processes like combustion. The modelling of a gas network focuses on the transport of gas through pipes based on mass flow and pressure.

This section gives a brief derivation of the mass flow of a gas in a pipe, based on [35], Chapter 4 of [41] and Chapter 2 of [33]. The section ends with a brief discussion on other common gas network elements.

2.2.1. Mass Flow in a Pipe

According to [41], there are many different models and formulas used to describe steady-state flow of gas through a pipe. The gas dynamics within a single pipe are described by a set of PDEs, often

called the Euler equations. For our use-case, assumptions have to be made, as we are interested in steady-state isothermal flow. This implies that there is no time-dependence and the gas temperatures are constant. Additionally, assumptions are made to simplify calculations and derivations. We thus assume negligible kinetic energy change in a pipe, constant compressibility of gas in a pipe, constant friction coefficient along the pipe length, validity of Darcy friction loss relationship across the pipe and horizontal pipes.

Under steady-state isothermal conditions, the governing Euler equations reduce to

$$\frac{\partial p}{\partial x} + f\frac{|v|v}{2D}\rho = 0, \tag{2.20}$$

where we have gas pressure $p(x)$ in pascal (Pa), dimensionless constant friction factor f, gas velocity $v(x)$ in $\mathrm{m\,s^{-1}}$, gas density $\rho(x)$ in $\mathrm{kg\,m^{-3}}$, and constant inner pipe diameter D in meter (m). Denote by $q(p_1, p_2)$ the constant *gas mass flow* in a pipe in $\mathrm{kg\,s^{-1}}$, depending on the pressure p_1 at the inlet of the pipe, and pressure p_2 at the outlet side of the pipe. Solving (2.20), as is done in detail in [41] and [33], results in the following expression of the mass flow,

$$q(p_1, p_2) = \mathrm{sign}(\Delta p) C^g \sqrt{\frac{|\Delta p|}{f}}, \tag{2.21}$$

with C^g the pipe constant defined differently for high-pressure and low-pressure systems as

$$C^g = \begin{cases} \frac{\pi}{8}\sqrt{\frac{SD^5}{TR_{\text{air}}LZ}}, & \text{High pressure,} \\ \frac{\pi}{8}\sqrt{\frac{2p_nSD^5}{T_nR_{\text{air}}L}}, & \text{Low pressure.} \end{cases} \tag{2.22}$$

The pressure drop Δp in (2.21) is also defined differently for high- and low-pressure systems as

$$\Delta p = \begin{cases} p_1^2 - p_2^2, & \text{High pressure,} \\ p_1 - p_2, & \text{Low pressure.} \end{cases} \tag{2.23}$$

In the pipe constant expression (2.22) we have S the dimensionless specific gravity of the gas, T the constant gas temperature in kelvin (K), R_{air} the gas constant of air in $\mathrm{J\,kg^{-1}\,K^{-1}}$, L the length of the pipe in meter (m), and Z the dimensionless compressibility factor.

For low-pressure systems we are able to make the following additional assumptions:

1. $Z \approx 1$: In a low pressure system the real gas is close to ideal gas.
2. $T \approx T_n$: In the low-pressure system we can assume that the temperature T is close to the standard temperature T_n.
3. $p_1^2 - p_2^2 \approx 2p_n(p_1 - p_2)$: Linearization around p_n, the standard pressure.

There are various models that one can use for the friction factor. It can be assumed that the friction is a constant, dependent on the pipe parameters or dependent on the gas mass flow, $f = f(q)$. Several models are discussed in Chapter 2 of [35] and Chapter 4 of [41].

2.2.2. Gas Network Elements

Besides pipes, there are also other important network elements in gas networks.

- Valves: Valves are used to route or block gas flow, and can be either closed or open. If the valve is open, we have $p_i = p_j$, $T_i = T_j$ and $q_{ij} \neq 0$. If the valve is closed the gas is prevented from flowing, which results in $q_{ij} = 0$ and p_i, p_j are decoupled.
- Control valves: Also known as *pressure regulators*, control valves are used to interconnect network parts operating at high pressure with the low pressure parts. If the valve is active, the pressure at the outlet can be reduced to a controllable value. As a result of the pressure drop,

the temperature of the gas also drops. If the control valve is closed, we have $q_{ij} = 0$ and p_i, p_j are decoupled. If the control valve is active, the pressure difference can be kept within a predetermined range around pressure p_{cv}, so we have $0 \leq p_{cv} - \epsilon \leq \Delta p_{ij} \leq p_{cv} + \epsilon$ and $q_{ij} \geq 0$.

- Compressor machines: Compressors are used to increase the pressure of the incoming gas to a higher outflow pressure. There are different types of compressor machines that can be some of the most complex parts of gas networks. A general compressor can be described by $p_j = rp_i$, with the constant r depending on the compressor, and $q_{ij} \geq 0$.

3

Modelling Single-Carrier Energy Networks

The aim of this chapter is to formulate the mathematical models of the single-carrier (SC) energy networks, focussing on the carriers electricity and gas. The contents and aim of this chapter are based on the work of Markensteijn [35]. The carrier-specific concepts, terms and definitions can be found in Chapter 2.

Section 3.1 describes how energy networks are represented as graphs, in a general setting as well as carrier-specific. Next, these graph formulations are used in Section 3.2 to derive, formulate and understand the steady-state load flow equations of electricity and gas networks.

3.1. Graph Representation of Energy Networks

Energy systems can be represented by graphs to visualize the connection between network components, producers of energy and consumers. This section discusses some general concepts from networks in graph theory to understand network modelling, based on Chapters 1 and 6 of [15]. Additionally, it discusses how electricity and gas networks can be interpreted as graphs, using notation, concepts and principles from [41] and [36].

3.1.1. Graphs and Networks

A *graph* $\mathcal{G}$ is a pair $(\mathcal{V}, \mathcal{E})$ of sets such that $\mathcal{E} \subseteq [\mathcal{V}]^2$. The set $\mathcal{V}$ is a set of *nodes*/*vertices* v_i, while $\mathcal{E}$ is a set of pairs of nodes $e_k = \{v_i, v_j\}$, referred to as *links*/*edges*. A graph is *undirected* if the links $\mathcal{E}$ form an unordered set of nodes. Conversely, a graph is *directed* if the links form an ordered set of nodes $e_k = (v_i, v_j)$, so that the link has a direction from node v_i to v_j. The *size*/*order* of a graph, denoted by $|\mathcal{V}|$, is the number of nodes of the graph. Similarly, let $|\mathcal{E}|$ denote the number of links.

The *incidence matrix* $A \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{E}|}$ describes the interconnection between nodes $\mathcal{V} = \{v_1, \dots, v_n\}$ and links $\mathcal{E} = \{e_1, \dots, v_m\}$ in a graph $\mathcal{G}$. If the graph is undirected, the incidence matrix is defined as

$$A_{ik} = \begin{cases} 1, & \text{link } e_k \text{ is connected to node } v_i \\ 0, & \text{otherwise.} \end{cases}$$

If the graph is directed, the incidence matrix is defined as

$$A_{ik} = \begin{cases} 1, & \text{link } e_k \text{ leaves node } v_i, \\ -1, & \text{link } e_k \text{ enters node } v_i, \\ 0, & \text{otherwise.} \end{cases} \tag{3.1}$$

A *terminal link* or *half link* is a special type of link that is only connected to one node and can be denoted by $t_l = \{v_i\}$. By definition, a terminal link can only be connected to one node, while a node can have

more than one terminal link connected to it. The set of terminal links t_l in a graph is denoted by $\mathcal{T}$ and the set of terminal links connected to node v_i is denoted by $\mathcal{T}(v_i)$.

A *network* is a graph with an associated physical model. The links of a network carry some kind of flow, such as water, electricity or data. In this research, we will focus on a network that represents the flow/transport of energy along its links, called an *energy (transportation) network*. Denote the energy network by $\mathcal{N} = \{\mathcal{G}, \mathcal{T}\} = \{\mathcal{V}, \mathcal{E}, \mathcal{T}\}$, a graph with a set of nodes and links, and a set of terminal links. Note that if graph $\mathcal{G}$ is (un)directed, then so is network $\mathcal{N}$.

The terminal links in an energy network are used to model inflow (source) and outflow (sinks) of energy. A node can represent such a source or sink, but since flow in a network is defined through links we associate the inflow and outflow of the sources with the terminal links. The terminal links thus represent the in- or outflow of the source or sink.

3.1.2. Electricity Network

A balanced alternating current (AC) power grid is represented by an undirected network, which we denote with $\mathcal{N}^e = \{\mathcal{V}^e, \mathcal{E}^e, \mathcal{T}^e\}$. In an electricity network a node, also called a *bus*, is an electrical component in the circuit. A link represents a transmission line. An example of an electricity network is given in Figure 3.1.

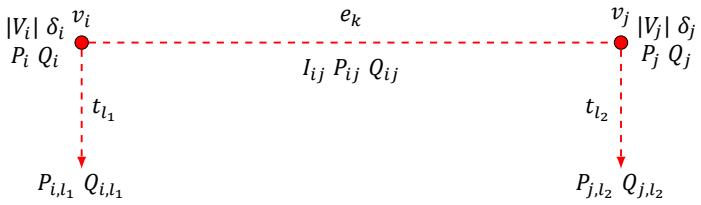


Figure 3.1: Electricity network with two nodes.

Generators and loads are represented by nodes and terminal links, and are modelled as *power injections*, which is the complex power supplied (consumed) by the generator (load). It consists of the injected real power P and injected reactive power Q. The injected real and reactive power through terminal link t_l supplied to node v_i are denoted by $P_{i,l}$ and $Q_{i,l}$, respectively. Note that multiple loads or generators can be connected to a single node. Therefore, define the *total real power* and *total reactive power* at node v_i as

$$P_i := \sum_l P_{i,l}, \quad Q_i := \sum_l Q_{i,l}, \tag{3.2}$$

respectively. Additionally, we denote the *real power flow* from node v_i to v_j with P_{ij}. Similarly, the *reactive power flow* Q_{ij} denotes the flow of reactive power between the two nodes. The convention is to denote consumed power as positive, and generated power as negative, following the direction of the terminal link arrows in Figure 3.1.

Additional variables of interest are the nodal voltage V and link current I. The voltage at node v_i is denoted as V_i. The current leaving node v_i and flowing to v_j is called I_{ij}. Conversely, the current leaving node v_j and flowing to v_i is denoted as I_{ji}. Note that if a transmission line does not contain any admittances, we have $I_{ij} = -I_{ji}$. Such a line is referred to as a lossless line. If there are series and/or shunt admittances present, we generally have $I_{ij} \neq -I_{ji}$ and the relation between I_{ij} and I_{ji} is determined via the transmission matrix (2.12), as described in Section 2.1.2. The *total injected current* at node v_i is defined as

$$I_i := \sum_l I_{i,l}.$$

The admittance matrix $\mathbf{Y}$ (2.16) or transmission matrix $\mathbf{T}$ (2.12) of the network describes the relation between the link current injected at the nodes and the nodal voltages. Note that the admittance matrix describes the currents in terms of voltages, whereas the transmission matrix describes sending current

and voltage in terms of receiving current and voltage. They can be used interchangeably for the modelling of a single link that connects two nodes. Using the admittance matrix $\mathbf{Y}$ or transmission matrix $\mathbf{T}$, we can express the link currents I_{ij} in terms of the nodal voltages V_i. As a result, the state of an electricity network can be fully described by four nodal parameters. For each node we thus describe:

- Voltage phasor magnitude $|V_i|$: The magnitude of the voltage phasor at node v_i,
- Voltage phasor angle δ_i: The angle of the voltage phasor at node v_i,
- Total real power P_i: The net generation or consumption of real power connected to node v_i,
- Total reactive power Q_i: The net generation or consumption of reactive power connected to node v_i.

There are different types of nodes/buses in a system. They represent different types of circuit elements and mathematically differ by which two of the four nodal parameters listed above are known. The three node types that occur in an electrical system are:

- **PQ bus**: Also known as a *load bus*, loads are modelled as constant power sinks. They do not have control over voltage levels, but their consumption is known through the total injected real and reactive powers P_i and Q_i. The unknown quantities are therefore the voltage magnitude $|V_i|$ and voltage phase angle δ_i. The number of PQ-buses in a system is denoted by $|\mathcal{PQ}|$.
- **PV bus**: At a PV bus, also called a *generator bus*, the total real power P_i supplied by a generator and the voltage magnitude $|V_i|$ are known. In a real-life power system, the voltage at a generator is usually kept within a predetermined range, which is required for the functioning and efficiency of the power system. The unknown quantities are therefore the reactive power Q_i and voltage phase angle δ_i. The number of PV-buses in a system is denoted by $|\mathcal{PV}|$.
- **Slack bus**: A slack bus is a special type of bus that acts as a reference point for the entire network. The voltage magnitude $|V_i|$ and voltage phase angle δ_i are specified. Within a power network the total injected power supplied, minus the total injected power consumed and total losses must balance to be zero. This balance is done by the slack node(s). The total injected real power P_i and injected reactive power Q_i are thus unknowns. The slack bus can be interpreted as a generator load that injects exactly the remaining power the system needs. The voltage magnitude $|V_i|$ and phase angle δ_i are known quantities and serve as a reference for the other voltages, meaning that we compute the voltage phase angles of the PQ and PV buses relative to the slack bus. There is typically only one slack bus in a system. However, in larger or specific systems there can be multiple slack buses. The number of slack buses in a system is denoted by $|\mathcal{S}|$.

An overview of electricity network node types can be found in Table 3.1.

Table 3.1: Electricity network node types

Node type	Specified	Unknown	Size notation
PQ or load bus	P, Q	$\|V\|, \delta$	$\|\mathcal{PQ}\|$
PV or generator bus	$P, \|V\|$	Q, δ	$\|\mathcal{PV}\|$
Slack bus	$\|V\|, \delta$	P, Q	$\|\mathcal{S}\|$

3.1.3. Gas Network

A gas system is represented by a directed network, which we denote with $\mathcal{N}^g = \{\mathcal{V}^g, \mathcal{E}^g, \mathcal{T}^g\}$. In a gas network a node can represent a sink, source or junction. The latter is an intersection of pipes where the gas is redistributed. A link can represent a pipe, compressor, valve and other gas network elements. An example of a gas network is given in Figure 3.2. The arrows in the directed graph show the direction of flow.

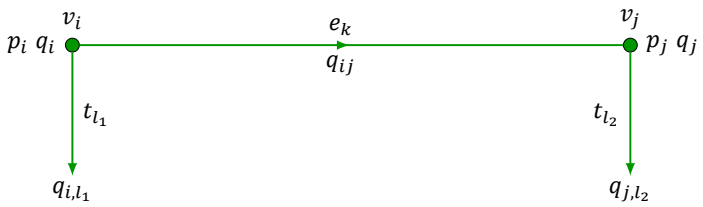


Figure 3.2: Gas network with two nodes.

The gas mass flow from a sink or source through terminal link t_l flowing to node v_i is denoted by $q_{i,l}$. Note that multiple sinks or sources can be connected to a single node. Therefore, the *total injected gas flow* at node v_i is defined as

$$q_i = \sum_l q_{i,l}. \tag{3.3}$$

The convention is to denote a supply of gas as negative and a sink as positive $q_{i,l}$, following the direction of the arrow in Figure 3.2.

We can describe a gas network with the following parameters:

- Nodal pressure p_i: The pressure at node v_i,
- Link gas flow q_{ij}: The gas mass flow from node v_i to node v_j.

There are different types of nodes in a gas network system. They represent different network elements and mathematically differ by which of the two nodal parameters are known.

- **Load node**: Loads are modelled as gas flow sinks. Their consumption of gas flow is known through the total injected gas mass flow q_i. The unknown quantity is therefore the pressure p_i. The number of load nodes in a system is denoted by $|\mathcal{Q}|$.
- **Slack node**: A slack node is a special type of node that acts as a reference point for the entire network. There is only one slack node in a system, and the pressure p_i is specified. Within a gas network, the total gas mass flow must balance to be zero. This balancing is done by the slack node where the total injected gas mass flow q_i is unknown. The slack bus can be interpreted as a generator or load that injects (consumes or supplies) exactly the remaining gas flow the system needs for a steady-state operation. The pressure p_i serves as a reference for the other nodal pressures. We therefore compute the pressure of the load nodes relative to the slack node. The number of slack nodes in a system is denoted by $|\mathcal{P}|$.

An overview of gas network node types can be found in Table 3.2.

Table 3.2: Gas network node types

Node type	Specified	Unknown	Size notation
Load node	q	p	$\|\mathcal{Q}\|$
Slack node	p	q	$\|\mathcal{P}\|$

3.2. Load Flow Equations

This section mathematically derives the load flow equations used in simulations of energy networks for electricity and gas networks, based on the energy network principles discussed in previous sections of this chapter and Chapter 2.

3.2.1. Load Flow Equations: Electricity Networks

A steady-state AC power grid is completely described by Kirchhoff's current law (KCL) (2.4), an equation linking the current and voltages through admittance (based on Ohm's law), and the complex power equation (2.9). These lead to the *load flow* or *power flow equations*. They relate the four quantities

described in Section 3.1.2 at all nodes $v_i \in \mathcal{V}^e$, by expressing nodal real power P_i and nodal reactive power Q_i in terms of system voltage magnitudes and phase angles.

In every node $v_i \in \mathcal{V}^e$, KCL (see Section 2.1.1) holds, stating that the sum of all injected currents must equal the sum of all outgoing currents. Following the notation used in Section 3.1.2, we can write KCL as

$$\sum_l I_{i,l} = I_i = -\sum_{j,j\neq i} I_{ij}, \tag{3.4}$$

the sum of all injected currents in node v_i, $I_{i,l}$, equals the sum of all currents leaving the node, I_{ij}, where the minus sign follows from the convention of choosing the direction of flow.

Additionally, the complex power equation (see Section 2.1.1) holds in every node. It states that in every node v_i, the product of the nodal voltage V_i and complex conjugate of the nodal current I_i can be expressed as a complex number that relates the real and reactive powers P_i and Q_i, so

$$\sum_l S_{i,l} = S_i = V_i(I_i)^* = P_i + \mathbf{i}Q_i,$$

with S_i the total injected complex power at node v_i.

Using Ohm's law, we can denote the current I_{ij} from node v_i to node v_j in terms of the voltages and total admittance. The result of this substitution is the admittance matrix (2.16) or the transmission matrix (2.12). The derivation of both matrices can be found in Section 2.1.2. Using the transmission matrix we can substitute (2.14) into KCL (3.4), resulting in

$$\begin{aligned} I_i &= -\sum_{j,j\neq i} \left(\Upsilon_{ii,j} V_i + \Upsilon_{ij} V_j\right), \\ &= -\sum_j Y_{ij} V_j. \end{aligned}$$

Here, Y_{ij} is a component of the admittance matrix $\mathbf{Y}$. Therefore, one can alternatively directly use the admittance matrix (2.16) in place of substitution of the transmission matrix into KCL to obtain this same result. We denote the real and complex parts of the admittance term $\mathbf{Y}_{ij}$ as $\mathrm{Re}(\mathbf{Y}_{ij}) = G_{ij}$ and $\mathrm{Im}(\mathbf{Y}_{ij}) = B_{ij}$. As a result, we can compute the complex conjugate of the current, I_i^*,

$$I_i^* = -\sum_j Y_{ij}^* V_j* = -\sum_j \left(G_{ij} - \mathbf{i}B_{ij}\right)\left(|V_j|e^{-\delta_j \mathbf{i}}\right).$$

Using this expression, the nodal complex power equation can be expressed in nodal real and reactive powers,

$$\begin{aligned} S_i &= V_i(I_i)^*, \\ &= -|V_i|e^{\delta_i \mathbf{i}}\left[\sum_j \left(G_{ij} - \mathbf{i}B_{ij}\right)\left(|V_j|e^{-\delta_j \mathbf{i}}\right)\right], \\ &= -|V_i|^2\left(G_{ii} - \mathbf{i}B_{ii}\right) - \sum_{j,j\neq i}\left(G_{ij} - \mathbf{i}B_{ij}\right)|V_i||V_j|e^{(\delta_i - \delta_j)\mathbf{i}}, \\ &= -\left[|V_i|^2 G_{ii} + \sum_{j,j\neq i}|V_i||V_j|\left(G_{ij}\cos\delta_{ij} + B_{ij}\sin\delta_{ij}\right)\right] - \\ &\quad \mathbf{i}\left[-|V_i|^2 B_{ii} - \sum_{j,j\neq i}|V_i||V_j|\left(B_{ij}\cos\delta_{ij} - G_{ij}\sin\delta_{ij}\right)\right], \\ &= P_i + \mathbf{i}Q_i, \end{aligned} \tag{3.5}$$

where $\delta_{ij} = \delta_i - \delta_j$. This result for the complex power equation leads us to the formulation of the load for equations for electricity networks.

Load Flow Equations
The load flow equations for an electricity network can be written as the following non-linear system of equations

$$\mathbf{F}^e(\mathbf{x}^e) = \mathbf{0}. \tag{3.6}$$

The argument $\mathbf{x}^e$ is defined as

$$\mathbf{x}^e = \begin{bmatrix} \boldsymbol{\delta} \\ |\mathbf{V}| \end{bmatrix}.$$

The vectors $\boldsymbol{\delta}$ and $\mathbf{V}$ contain the unknown nodal voltage angles and magnitudes. The number of unknown voltage angles is $|\mathcal{V}^e| - |\mathcal{S}|$, as the slack nodes have a known voltage angle. Similarly, the number of unknown voltage magnitudes is $|\mathcal{V}^e| - |\mathcal{S}| - |\mathcal{PV}|$, as the slack nodes and PV nodes have prescribed voltage magnitude. The vector $\mathbf{x}^e$ thus has $2(|\mathcal{V}^e| - |\mathcal{S}|) - |\mathcal{PV}|$ entries.

The function $\mathbf{F}^e(\mathbf{x}^e)$ is defined as

$$\mathbf{F}^e(\mathbf{x}^e) = \begin{bmatrix} \mathbf{F}^P(\mathbf{x}^e) \\ \mathbf{F}^Q(\mathbf{x}^e) \end{bmatrix}. \tag{3.7}$$

The vectors $\mathbf{F}^P(\mathbf{x}^e)$ and $\mathbf{F}^Q(\mathbf{x}^e)$ contain the complex power equation for the nodes with known injected real and reactive power, respectively. The number of nodes with known total real power is $|\mathcal{PV}| + |\mathcal{PQ}|$, whereas the number of nodes with known total reactive power is $|\mathcal{PQ}|$. The vector $\mathbf{F}^e(\mathbf{x}^e)$ thus has $|\mathcal{PV}| + 2|\mathcal{PQ}|$ components. Suppose that the $\tilde{i}$-th entry of $\mathbf{F}^P(\mathbf{x}^e)$ corresponds to node v_{i_1}, which has known total real power, and that the $\hat{i}$-th entry of $\mathbf{F}^Q(\mathbf{x}^e)$ corresponds to node v_{i_2}, which has known total real power, then the entries of $\mathbf{F}^e(\mathbf{x}^e)$ are determined using the result in (3.5),

$$\begin{aligned} \mathbf{F}^P_{\tilde{i}}(\mathbf{x}^e) &= P_{i_1} + |V_{i_1}|^2 G_{i_1 i_1} + \sum_{j, j \neq i_1} |V_{i_1}||V_j| \left(G_{i_1 j} \cos \delta_{i_1 j} + B_{i_1 j} \sin \delta_{i_1 j} \right), \\ \mathbf{F}^Q_{\hat{i}}(\mathbf{x}^e) &= Q_{i_2} - |V_{i_2}|^2 B_{i_2 i_2} - \sum_{j, j \neq i_2} |V_{i_2}||V_j| \left(B_{i_2 j} \cos \delta_{i_2 j} - G_{i_2 j} \sin \delta_{i_2 j} \right). \end{aligned} \tag{3.8}$$

Recall that P_{i_1} and Q_{i_2} are defined as the sum of injected powers, (3.2).

Note that $|\mathcal{V}^e| = |\mathcal{PQ}| + |\mathcal{PV}| + |\mathcal{S}|$, so $|\mathcal{PV}| + 2|\mathcal{PQ}| = 2(|\mathcal{V}^e| - |\mathcal{S}|) - |\mathcal{PV}|$. The total system (3.6) thus contains $|\mathcal{PV}| + 2|\mathcal{PQ}|$ equations with $|\mathcal{PV}| + 2|\mathcal{PQ}|$ unknown variables.

3.2.2. Load Flow Equations: Gas Networks

A steady-state gas network can be completely described by conservation of mass and link equations that model the network elements. This leads to the *load flow equations* that relate the nodal pressures and total injected gas flow described in Section 3.1.3 at all nodes v_i.

In every node $v_i \in \mathcal{V}^g$, conservation of mass must hold. Therefore, the sum of all flows going into node v_i and all flows going out of node v_i must balance to be zero. This leads to

$$\sum_{j, j \neq i} q_{ji} - \sum_{j, j \neq i} q_{ij} - q_i = 0. \tag{3.9}$$

The first term corresponds to gas flowing into node v_i from other connected nodes, the second corresponds to the gas flowing out of node v_i to other connected nodes, and the final term q_i is the total injected gas flow (3.3). The signs follow from the convention of choosing the direction of flow.

Additionally, each link has a link equation, which differs per gas network element. If a link represents a pipe, then the results of Section 2.2.1 can be used, leading to the pipe link equation. For the pipe that allows for gas mass flow between nodes v_i and v_j, we have pipe link equation

$$q_{ij}(p_i, p_j) = \text{sign}(\Delta p_{ij}) C^g \sqrt{\frac{|\Delta p_{ij}|}{f}}, \tag{3.10}$$

where Δp_{ij} represents the pressure difference between the nodes v_i and v_j, defined in (2.23), and constant C^g is given by (2.22). Alternatively, instead of expressing the gas mass flow in terms of pressure difference as $q(\Delta p)$, one can express the pressure difference in terms of gas mass flow as $\Delta p(q)$. This leads to equivalent pipe link equation

$$\Delta p_{ij} = (C^g)^{-2} f |q_{ij}| q_{ij}. \tag{3.11}$$

Load Flow Equations

The load flow equations for a gas network can be written as the following system of equations

$$\mathbf{F}(\mathbf{x}^g) = \mathbf{0}. \tag{3.12}$$

There are various formulations used for the load flow equations of a gas network. All describe the same network but define $\mathbf{F}$ and $\mathbf{x}^g$ differently. The formulations used are the nodal, loop, nodal-loop and full formulations. In the nodal formulation there is a substitution of link equations (3.10) into the conservation of mass equations (3.9), as is done in [41]. The unknown quantities then consist of the unknown total injected gas flows and the unknown nodal pressures. However, one can only substitute link equations into conservation of mass if the link equations contain the variable q_{ij}. If the link represents a pipe, this is the case, (see (3.10)), but if the link represents a different network element, such as a compressor, the gas mass flow term is not necessarily present in the equation, see Section 2.2.2. The nodal formulation is therefore only used where all links represent pipes. In the full formulation, there is no substitution. The unknown quantities consist of the unknown nodal pressures, the unknown total injected gas flows and gas mass flows along all links. This formulation is more widely applicable as it can include all types of gas network elements. Therefore, the full formulation is considered in the remainder of this research.

In the full formulation, the argument $\mathbf{x}^g$ is defined as

$$\mathbf{x}^g = \begin{bmatrix} \mathbf{q} \\ \mathbf{p} \end{bmatrix}.$$

The vector $\mathbf{q}$ contains the unknown link flows. Since all link flows are unknown, this vector is of length $|\mathcal{E}^g|$. The vector $\mathbf{p}$ consists of all unknown nodal pressures. The number of unknown nodal pressures is $|\mathcal{V}^g| - |\mathcal{P}| = |\mathcal{Q}|$, the total number of nodes minus the number of slack nodes with specified nodal pressure. The vector $\mathbf{x}^g$ thus has $|\mathcal{E}^g| + |\mathcal{Q}|$ entries.

The function $\mathbf{F}^g(\mathbf{x}^g)$ is defined as

$$\mathbf{F}^g(\mathbf{x}^g) = \begin{bmatrix} \mathbf{F}^q(\mathbf{x}^g) \\ \mathbf{F}^L(\mathbf{x}^g) \end{bmatrix}. \tag{3.13}$$

The vector $\mathbf{F}^q(\mathbf{x}^g)$ is the vector of conservation of mass, and contains the components as in (3.9) for only the nodes with known total injected gas flow. Suppose that the $\tilde{i}$-th entry of $\mathbf{F}^q(\mathbf{x}^g)$ corresponds to node v_i, which has known total injected gas flow q_i, then we have

$$\mathbf{F}^q_{\tilde{i}}(\mathbf{x}^g) = \sum_{j, j \neq i} q_{ji} - \sum_{j, j \neq i} q_{ij} - q_i.$$

Note that this is a linear relation of components and we can therefore write conservation of mass as a linear system using the directed incidence matrix of the network as defined in (3.1). Since we are only considering the conservation of mass equation for nodes with known total injected gas flow, the *reduced* incidence matrix $\mathbf{A}^{\mathbf{g}'}$ of the network is used, which only contains the rows of the incidence matrix corresponding to the nodes with known total injected gas flow. This results in the linear system

$$\mathbf{F}^q = \mathbf{A}^{\mathbf{g}'}\mathbf{q} - \mathbf{q}^{\text{inj}}, \tag{3.14}$$

where $\mathbf{q}^{\text{inj}}$ is the vector with all known total injected gas flows. The vector $\mathbf{F}^q$ thus has length $|\mathcal{Q}|$.

The vector $\mathbf{F}^L(\mathbf{x}^g)$ is the vector of unknown link flows. If the network consists of only pipes, the link

equations are given by (3.10) or (3.11). In this derivation, we will use (3.11), as this formulation is generally more robust when linearising and solving the load flow equations. Additionally, for a low pressure system, (3.11) results in an equation that is linear in the pressures. Suppose that the k-th entry of $\mathbf{F}^L(\mathbf{x}^g)$ corresponds to a link (gas pipe) from node v_i to node v_j, then we have

$$\mathbf{F}_k^L(\mathbf{x}^g) = \Delta p_{ij} - (C^g)^{-2} f|q_{ij}|q_{ij}. \tag{3.15}$$

The vector $\mathbf{F}^L(\mathbf{x}^g)$ is of length $|\mathcal{E}^g|$, as we assume all link flows are unknown.

The total system (3.13) thus contains $|\mathcal{Q}| + |\mathcal{E}^g|$ equations with $|\mathcal{Q}| + |\mathcal{E}^g|$ unknowns.

4

Modelling Multi-Carrier Energy Networks

Different energy carriers such as electricity, gas and heat, can interact with each other, leading to one combined system, referred to as a multi-energy system (MES). Such a system can be represented by a multi-carrier energy network (MCN) by coupling nodes of different carriers. An integrated network can be used to more realistically model and analyse the energy grid as a whole.

This chapter discusses various methods of coupling that can be used for an MCN in Section 4.1, and includes the formulation introduced in [35]. The integrated load flow equations and node types of an MES can be found in Section 4.2.

4.1. Coupling

The coupling of single-carrier (SC) energy networks to form an MCN is a key element of integrated network modelling. This section first briefly introduces types of coupling units that convert energy from one carrier to another. These coupling units can be included in an MCN in various ways, which is discussed next. Finally, the coupling method proposed by Markensteijn [35] is described in more detail.

4.1.1. Coupling Units

In an MES, the SC systems interact with each other through various elements, called *coupling units*. These elements represent conversion units that transfer energy from one carrier to one or more other carriers. Table 4.1 shows an overview of conversion units in an MES that transform energy from one carrier to another. A combined heat and power plant (CHP) makes use of the residual heat that might occur during energy conversion in a coupling unit. As a result, less energy is lost during conversion.

Table 4.1: Conversion units and examples in multi-carrier energy networks

Input carrier	Output carriers	Abbreviation	Example
Electricity	Gas	P2G	Electrolyser
Electricity	Heat	E2H	Electric boiler
Gas	Electricity	GFG	Gas-fired generator
Gas	Heat	G2H	Gas boiler
Gas	Electricity, heat	CHP	Combined heat and power plant

When a gas is combusted, it releases energy. A gas-specific property is the gross heating value (GHV), that represents how much energy is released by complete combustion. Additionally, when energy is transformed from one carrier to another, there is likely to be a loss of energy. Therefore, the efficiency term η is introduced, where $0 < \eta < 1$ indicates how much energy is actually transferred from one carrier to another.

In coupled gas-electricity networks, two types of coupling units can occur, which are the power-to-gas (P2G) and gas-fired generator (GFG). The coupling unit determines the direction of flow of energy

as well as the way in which the energy is transformed. In this research, a GFG unit is modelled as

$$P = \eta * (\text{GHV} * q), \tag{4.1}$$

where P is the electrical active power leaving the coupling unit, and q the gas mass flow entering. This equation indicates that the gas is combusted, releasing a total energy indicated by GHV $* q$. The number of energy that is then actually transformed into active power P is computed via efficiency η. Similarly, a P2G unit is modelled as

$$q = \frac{\eta * P}{\text{GHV}}, \tag{4.2}$$

where P is the active power entering the coupling unit, and q the gas mass flow leaving. This equation indicates that the active power supplied by the electrical component is transformed into energy using efficiency η, which is then transformed into gas mass flow via its GHV.

4.1.2. Methods for Coupling Single-Carrier Networks

As described in [35], there are three main methods generally used to couple different energy carriers into an MES:

1. Coupling link: Connect two nodes of a different energy carrier through a coupling link, such that this coupling link represents the coupling unit. This method has no straightforward interpretation of a coupling link and it is less intuitive how more than two different energy carriers are coupled.
2. Merged node: Merge two nodes of a different carrier into one node, such that the incoming carrier, outgoing carrier and coupling unit are represented by one node. This method has the practical downside that nodal parameters might have to be doubly defined. Additionally, it has to be possible to differentiate between links of different carriers connected to the merged node in order to apply Kirchhoff's law in constructing the load flow equations.
3. Coupling nodes and dummy links: Introduce an additional coupling node and connect the two different energy carrier nodes with dummy links to the coupling node, such that the coupling node represents the coupling unit. With this method the models of the SC parts are only slightly altered and the connection between the carriers is shown explicitly.

In the rest of this report the coupling of SC parts is considered to be done with coupling nodes and dummy links, and is further discussed in Section 4.1.3.

4.1.3. Coupling Nodes and Dummy Links

The *coupling node* represents an MES coupling unit as described in Section 4.1.1 that couples at least two energy carriers. An example of a graph representation of an MES with one coupling node is given in Figure 4.1. By definition, no nodal parameters are associated with the coupling node and it is not part of any of the SC parts, but rather belongs to its own coupling part. If a coupling node couples nodes of the same carrier, it is referred to as *homogeneous*. Such a node can be introduced when, for example, coupling high-voltage and low-voltage electricity networks. A network with only homogeneous (coupling) nodes is a homogeneous network. Conversely, if a coupling node couples nodes of different carriers, it is referred to as *heterogeneous*. A network with at least one heterogeneous coupling node is a heterogeneous network. The total number of coupling nodes in an MES is denoted by $\mathcal{V}^c$.
The different SC parts in an MES are connected to the coupling node via *dummy links*. They only represent connections between the SC parts and therefore do not represent physical system components. However, they have the same parameter types associated with them as the links of the SC network they are connected to and could therefore be interpreted as lossless links. Gas and electricity dummy links are interpreted as follows:

- Gas dummy link: A gas dummy link only has a gas mass flow q along the link as link parameter, but not a link equation.
- Electricity dummy link: An electricity dummy link has a current I and real and reactive powers P and Q associated with it. Since the dummy link can be interpreted as a lossless power line, it must therefore hold that $I_{ij} = -I_{ji}$ as well as that $P_{ij} = -P_{ji}$ and $Q_{ij} = -Q_{ji}$.

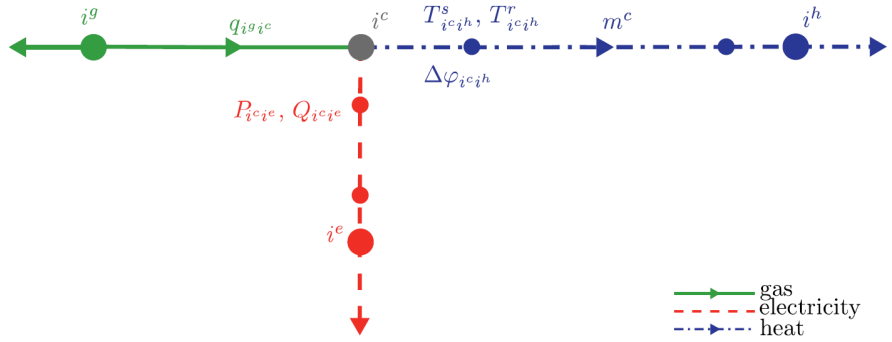


Figure 4.1: Taken from [35]: '*Network representation of a coupling node, showing quantities of interest for LF and the network elements they are associated with.*'

Following the introduction of dummy links, the MES can be easily separated in SC parts by transforming the dummy links into terminal links. This flexibility allows for general application to MESs.

4.2. Load Flow Equations

Using the coupling method described in Section 4.1, this section describes the mathematical formulation of an MCN. The newly introduced coupling equations and occurrences of new node types lead to the load flow equations of an MCN. In the remainder of this research, only coupled electricity and gas networks will be considered.

4.2.1. Coupling Equations

Denote an MCN with $\mathcal{N} = \{\mathcal{V}, \mathcal{E}, \mathcal{T}\}$, where $\mathcal{V} = \{\mathsf{U}_{x\in X}\, \mathcal{V}^x, \mathcal{V}^c\}$, $\mathcal{E} = \{\mathsf{U}_{x\in X}\, \mathcal{E}^x, \mathcal{E}^c\}$ and $\mathcal{T} = \{\mathsf{U}_{x\in X}\, \mathcal{T}^x\}$ with included SC energy networks X. The set $\mathcal{V}^c$ contains all coupling nodes, while $\mathcal{E}^c$ is the set of dummy links. Note that there are no terminal links associated with the coupling, as coupling nodes don't have terminal links associated to them.

In every heterogeneous coupling node, one or more (non-)linear coupling equations hold that represent the conversion of energy from one carrier to the other. The general form of the nodal coupling equations associated with coupling node $v_i \in \mathcal{V}^c$ connected to homogeneous node v_j is given as

$$\mathbf{F}_i^c(\mathbf{x}^c) = 0. \tag{4.3}$$

Note that one coupling unit can result in one or more coupling equations. The coupling parameters for coupled electricity and gas networks are given as

$$\mathbf{x}^c = \begin{bmatrix} \mathbf{q}_c \\ \mathbf{P}_c \\ \mathbf{Q}_c \end{bmatrix}, \tag{4.4}$$

where $\mathbf{q}_c$ is the vector of gas mass flows of the gas dummy links, and $\mathbf{P}_c$ and $\mathbf{Q}_c$ are the real and reactive power of the electricity dummy links.

In this research, the function $\mathbf{F}^c(\mathbf{x}^c)$ consists of GFG and P2G equations, given in (4.1) and (4.2), respectively. This gives

$$\mathbf{F}^c_{\mathsf{GFG}}(\mathbf{q}_c, \mathbf{P}_c) = P_c - \eta(\mathsf{GHV} q_c), \tag{4.5}$$

for a GFG unit. For a P2G unit we get

$$\mathbf{F}^c_{\mathsf{GFG}}(\mathbf{q}_c, \mathbf{P}_c) = \eta P_c - \mathsf{GHV} q_c. \tag{4.6}$$

Note that $\mathbf{Q}_c$ does not appear in the equations for a GFG or P2G unit. In the remainder of this research, the $\mathbf{Q}_c$ therefore does not occur.

4.2.2. Additional Node Types

Generally, the coupling of SC networks results in more added variables than added equations. A necessary condition for a well-posed linear system is to have as many variables as equations, called a square system. The load flow equations of the electricity and gas networks are generally non-linear. As a result of the non-linearity, it is no longer necessary to have as many unknown variables as equations for the system to be well-posed. However, these non-linear systems are often linearised during the solution process, such as in the Newton-Raphson (NR) method. A necessary condition for the existence of the inverse of the system Jacobian in NR is that it must be square. It is thus generally desirable for the non-linear system of equations for an MES to be square, by having as many unknown variables as equations. In order for the multi-carrier load flow problem to have as many equations as unknown parameters, we require additional parameters to be known beforehand. More practically, the interpretations of the scenarios modelled using coupling nodes often naturally result in more known parameters in order to solve the load flow problem. For example, parameters of the slack node in one SC network can be replaced by the coupling parameters, essentially using the other coupled SC network as the slack. Consequently, new node types are introduced that only occur in MCNs.

Possible gas and electricity node types in an MCN are given in Table 4.2 and 4.3, respectively. Mathematically, more electrical node types are possible than indicated in the table, by combining different known and unknown parameters. However, the node types mentioned in Table 4.2 and 4.3 are the node types that are generally physically possible in an MES with an electrical component. As a result of the additional node types, and thus a possible transfer of the slack function from one carrier to another, it can occur that the load flow problem of one SC network is overdetermined, while the load flow problem of another SC network is underdetermined. However, the final MCN has as many equations as unknowns.

Table 4.2: Gas node types in a multi-carrier network

Node type	Specified	Unknown
Load node	q	p
Slack node	p	q
Reference load node	p, q	-
Slack node	-	p, q

Table 4.3: Electricity node types in a multi-carrier network

Node type	Specified	Unknown
PQ or load bus	P, Q	$\lvert V\rvert, \delta$
PV or generator bus	$P, \lvert V\rvert$	Q, δ
Slack bus	$\lvert V\rvert, \delta$	P, Q
PQV bus	$P, Q, \lvert V\rvert$	δ
QVδ bus	$Q, \lvert V\rvert, \delta$	P
PQVδ bus	$P, Q, \lvert V\rvert, \delta$	-

4.2.3. Load Flow Equations: Multi-Energy Systems

To construct the load flow equations of a coupled network, one first constructs the load flow equations of the SC parts according to carrier-specific formulation. The load flow equations for electricity and gas are described in Sections 3.2.1 and 3.2.2, respectively. During the construction of the SC components, the dummy link variables are taken into account, whereas the coupling nodes are not. The load flow equation(s) for the coupling nodes follow from the specific unit used for coupling and are of general form (4.3). Combining the load flow equations of the individual electricity and gas networks and the coupling nodes results in the following MES load flow problem:

$$\mathbf{F}(\mathbf{x}) = \begin{bmatrix} \mathbf{F}^g(\mathbf{x}^g, \mathbf{x}^c) \\ \mathbf{F}^e(\mathbf{x}^e, \mathbf{x}^c) \\ \mathbf{F}^c(\mathbf{x}^c) \end{bmatrix} = \mathbf{0}, \quad \mathbf{x} = \begin{bmatrix} \mathbf{x}^g \\ \mathbf{x}^e \\ \mathbf{x}^c \end{bmatrix}. \tag{4.7}$$

The load flow equations for the gas network in the multi-carrier system are included in $\mathbf{F}^g$ that depend on the gas network parameters included in $\mathbf{x}^g$ (the nodal pressures and link gas mass flows), as well as on the dummy link gas mass flows included in $\mathbf{x}^c$. Similarly, the load flow expressions corresponding to the electricity network are included in $\mathbf{F}^e$ that depend on the electricity network parameters in $\mathbf{x}^e$, which are the nodal voltage magnitudes and angles. Additionally, electricity load flow also depends on the real and reactive powers of the dummy links included in $\mathbf{x}^c$. Finally, the nodal coupling expressions are included in $\mathbf{F}^c$ that model the coupling units. The coupling equations only depend on the dummy link coupling variables $\mathbf{x}^c$.

4.3. Solution Methods for Multi-Carrier Energy Systems

Studies on hybrid energy flow of MESs often focus on the establishment of models and formulations [62, 65], while only a few studies are dedicated to the solution methods used in such load flow calculations [44]. Several studies on smaller MCNs make use of NR, combined with direct solvers for the linear systems [37, 62]. Solving large integrated energy networks all at once remains an open challenge [39].

4.3.1. Decomposed Methods

Some studies on solving MESs investigate the possibility of decoupling a coupled MES, and using direct solvers to solve the smaller or simpler sub-systems, categorised as decomposed methods.

Markensteijn [35] formulated a decoupled approach to solve load flow problems of coupled MCNs. Through the use of interface conditions (IFCs), she describes how a coupled network can be formulated such that the SC networks can be solved separately. The blocks in the Jacobian of a coupled network are generally not square. In her decoupled formulation, the SC sub-matrices in the Jacobian are made square by including information obtained from the coupling in the SC subnetwork. It is mentioned that the decoupled formulation is equivalent to a permutation of the Jacobian matrix of the coupled system, as in (5.7). Markensteijn states that when the Jacobian matrix of a coupled network is permuted such that the SC sub-matrices are square, domain decomposition (DD) or block iterative methods can be used to solve the linear system.

Zhang et al. [64] introduced a partial decoupling method for fast calculation of energy flow in large-scale heat and electricity integrated energy systems, which operate in a heat-load-following mode. Although the focus of the study is on the partial decoupling method for a district heating system (DHS), they also solve a coupled electro-thermal system. Since they consider a coupled system in heat-load-following mode, their methodology of solving the coupled system consists of first solving the DHSs iteratively, then updating the electric power of the combined heat and power (CHP) units, and finally calculating the electric power flow using NR combined with a direct solver. The errors in the partial decoupling method for the DHS have little effect on the convergence of NR for the coupled electrical network, which is desirable.

4.3.2. Integrated Methods

Jia et al. [26] conducted an investigation into problems of the integrated method for electro-thermal energy systems. The integrated method fails in exceptional cases when energy reverses during iterations. Additionally, it is sensitive to the choice of initial values.

Qi, Li, and Bie [44] studied the inexact NR method with preconditioned generalized minimum residual (GMRES) for hybrid energy flow calculation of integrated energy systems, using an electro-thermal coupling system as a test case [44]. Their results show a significant acceleration effect on the calculation speed compared to classical methods, and it is therefore recommended to further investigate (preconditioned) iterative solvers applied to integrated energy systems.

5

Linear and Non-Linear Solution Methods

An energy network can be represented by a system of load flow equations. The systems of equations for electricity and gas networks are described in Chapter 3. The systems of equations for a multi-energy system (MES) where different energy carriers are combined are described in Chapter 4. In order to solve the energy flow problems they represent, these systems of, generally non-linear, equations have to be solved.

A general energy load-flow problem is represented by a non-linear system of equations. The most commonly used principle in solving non-linear systems is to linearize and use an appropriate linear solution method. The non-linear solution method Newton-Raphson (NR) is therefore introduced in Section 5.1. The linearization step includes the computation of the energy network Jacobian matrix, whose structure is influential in solver performance and convergence. Therefore, the general Jacobian matrix structures of electricity, gas and multi-carrier energy networks (MCNs) are also included.

A general linear system of equations can be written as a matrix-vector product

$$A\mathbf{x} = \mathbf{f}, \tag{5.1}$$

with $A \in \mathbb{R}^{n \times n}$ a non-singular coefficient matrix, $\mathbf{f} \in \mathbb{R}^n$ a right-hand side or forcing vector and $\mathbf{x} \in \mathbb{R}^n$ the unknown solution vector. The goal is to solve this system reliably and efficiently. Generally, a direct computation of the inverse A^{-1} is computationally expensive and inefficient, especially for larger systems. Different solution methods have been developed to determine the solution vector $\mathbf{x}$ without directly computing the inverse. They are either direct methods or iterative methods. In a direct solution method the coefficient matrix A is factorized into a product of matrices in such a way that computation of solution vector $\mathbf{x}$ is simplified and less expensive. Direct solution methods that are used in solving MES problems can be found in Section 5.2. The second class of solution methods is based on an iterative process that aims to improve an approximation of the solution vector $\mathbf{x}$ or minimise errors during each iteration. Such iterative methods are often used for large systems due to their efficiency compared to direct solution methods. In [24] and [12], it is concluded that an iterative method is the optimal method for large-scale power systems. Iterative methods that are used in solving MES problems can be found in Section 5.3. Finally, Section 5.4 introduces and discusses domain decomposition (DD) methods, based on the principle of divide-and-conquer. Preconditioning is another important aspect of linear solution methods. These techniques are discussed in the next chapter, Chapter 6.

5.1. Non-Linear Solution Methods

The systems of equations that describe steady-state load flow of energy networks are generally non-linear. A commonly used principle in solving these non-linear equations is to linearize and use a direct or iterative linear solution method, as described in Section 5.2 and Section 5.3. This section introduces the NR as well as the inexact NR method. Additionally, the general Jacobian matrices of electricity, gas and coupled networks are derived.

5.1.1. Newton-Raphson

The *NR* method is used to iteratively solve systems of non-linear equations of the form

$$\mathbf{F}(\mathbf{x}) = \mathbf{0}.$$

The *Jacobian matrix* J of $\mathbf{F}(\mathbf{x})$, $\mathbf{F} : \mathbb{R}^N \to \mathbb{R}^N$, $\mathbf{x} \in \mathbb{R}^N$ is given by

$$J_F = \nabla \mathbf{F} = \begin{bmatrix} \frac{\partial F_1}{\partial x_1} & \cdots & \frac{\partial F_1}{\partial x_N} \\ \vdots & \ddots & \vdots \\ \frac{\partial F_N}{\partial x_1} & \cdots & \frac{\partial F_N}{\partial x_N} \end{bmatrix}.$$

The NR method is based on a first-order multivariate Taylor expansion around iterate $\mathbf{x}_k$, such that for the next iterate $\mathbf{x}_{k+1}$ we have

$$\mathbf{F}(\mathbf{x}_{k+1}) = \mathbf{F}(\mathbf{x}_k) + J_F(\mathbf{x}_k)(\mathbf{x}_{k+1} - \mathbf{x}_k) + \text{h.o.t.}$$

The aim is to find $\mathbf{x}_{k+1}$ such that $\mathbf{F}(\mathbf{x}_{k+1})$ gets close to zero. The above equation therefore reduces to

$$J_F(\mathbf{x}_k)\mathbf{s}_k \approx -\mathbf{F}(\mathbf{x}_k),$$

where $\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k$, referred to as the update. The NR update equation is thus given as

$$J_F(\mathbf{x}_k)\mathbf{s}_k = -\mathbf{F}(\mathbf{x}_k). \tag{5.2}$$

During each iteration, this linear system is solved using a linear solution method (direct or iterative), after which the new solution approximation $\mathbf{x}_{k+1}$ is determined using the computed update, $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{s}_k$.

The class of *Inexact NR* methods compute an approximate solution to the Newton equations (5.2) in some unspecified manner such that

$$\frac{||\mathbf{r}_k||}{||\mathbf{F}(\mathbf{x}_k)||} \leq \eta_k. \tag{5.3}$$

Here $\mathbf{r}_k = J_F(\mathbf{x}_k)\mathbf{s}_k + \mathbf{F}(\mathbf{x}_k)$ is the residual, $||\mathbf{r}_k||/||\mathbf{F}(\mathbf{x}_k)||$ the relative residual and η_k a non-negative *forcing sequence* used to control the level of accuracy [13, 44, 23]. Under the assumption that the forcing sequence is uniformly less than one, all such inexact NR methods are locally convergent and exhibit the same quadratic convergence as the NR method [13].

5.1.2. Jacobian Matrices of Energy Networks

The Jacobian matrix of a (non-)linear system of equations is fundamental in the application of NR. It is insightful to investigate the general form of the Jacobian matrices of energy networks, since their spectral properties influence the convergence and accuracy of NR and the linear solution methods. This section gives the general form of the Jacobian matrix of electricity, gas and coupled networks, using the load flow equations of Section 3.2.1, Section 3.2.2 and Section 4.2, respectively.

Jacobian matrix electricity network

The Jacobian matrix of an electricity network based on steady-state load flow (3.7) is given as

$$J_{F^e} = \begin{pmatrix} \frac{\partial \mathbf{F}^P}{\partial \boldsymbol{\delta}} & \frac{\partial \mathbf{F}^P}{\partial |\mathbf{V}|} \\ \frac{\partial \mathbf{F}^Q}{\partial \boldsymbol{\delta}} & \frac{\partial \mathbf{F}^Q}{\partial |\mathbf{V}|} \end{pmatrix}. \tag{5.4}$$

Note that in (3.5), the load flow equations contain the components

$$\mathcal{P}_{ij} = |V_i||V_j| \left(G_{ij} \cos \delta_{ij} + B_{ij} \sin \delta_{ij}\right), \quad \mathcal{Q}_{ij} = -|V_i||V_j| \left(B_{ij} \cos \delta_{ij} - G_{ij} \sin \delta_{ij}\right).$$

Therefore, we can write the non-linear steady-state load flow equations for an electricity network as

$$\mathbf{F}_i^P(\mathbf{x}^e) = P_i + \sum_j \mathcal{P}_{ij}(\mathbf{x}^e) = P_i + |V_i|^2 G_{ii} + \sum_{j, j \neq i} \mathcal{P}_{ij}(\mathbf{x}^e) = 0,$$

$$\mathbf{F}_i^Q(\mathbf{x}^e) = Q_i + \sum_j \mathcal{Q}_{ij}(\mathbf{x}^e) = Q_i - |V_i|^2 B_{ii} - \sum_{j, j \neq i} \mathcal{Q}_{ij}(\mathbf{x}^e) = 0.$$

The Jacobian (5.4) then contains the components

$$\begin{aligned}
\frac{\partial \mathbf{F}_i^P}{\partial \delta_j} &= -|V_i||V_j|\left(B_{ij}\cos\delta_{ij} - G_{ij}\sin\delta_{ij}\right) && = Q_{ij},\\
\frac{\partial \mathbf{F}_i^P}{\partial \delta_i} &= \sum_{j,j\neq i}|V_i||V_j|\left(B_{ij}\cos\delta_{ij} - G_{ij}\sin\delta_{ij}\right) && = -\sum_{j,j\neq i} Q_{ij} = -Q_i - |V_i|^2 B_{ii},\\
\frac{\partial \mathbf{F}_i^Q}{\partial \delta_j} &= -|V_i||V_j|\left(G_{ij}\cos\delta_{ij} + B_{ij}\sin\delta_{ij}\right) && = -\mathcal{P}_{ij},\\
\frac{\partial \mathbf{F}_i^Q}{\partial \delta_i} &= \sum_{j,j\neq i}|V_i||V_j|\left(G_{ij}\cos\delta_{ij} + B_{ij}\sin\delta_{ij}\right) && = \sum_{j,j\neq i}\mathcal{P}_{ij} = P_i - |V_i|^2 G_{ii},\\
\frac{\partial \mathbf{F}_i^P}{\partial |V_j|} &= |V_i|\left(G_{ij}\cos\delta_{ij} + B_{ij}\sin\delta_{ij}\right) && = \frac{\mathcal{P}_{ij}}{|V_j|},\\
\frac{\partial \mathbf{F}_i^P}{\partial |V_i|} &= 2|V_i|G_{ii} + \sum_{j,j\neq i}|V_j|\left(G_{ij}\cos\delta_{ij} + B_{ij}\sin\delta_{ij}\right) && = \frac{\sum_{j,j\neq i}\mathcal{P}_{ij}}{|V_i|} + 2|V_i|G_{ii} = \frac{P_i + |V_i|^2 G_{ii}}{|V_i|},\\
\frac{\partial \mathbf{F}_i^Q}{\partial |V_j|} &= |V_i|\left(B_{ij}\cos\delta_{ij} - G_{ij}\sin\delta_{ij}\right) && = \frac{Q_{ij}}{|V_j|},\\
\frac{\partial \mathbf{F}_i^Q}{\partial |V_i|} &= -2|V_i|B_{ii} - \sum_{j,j\neq i}|V_j|\left(B_{ij}\cos\delta_{ij} - G_{ij}\sin\delta_{ij}\right) && = \frac{\sum_{j,j\neq i}Q_{ij}}{|V_i|} - 2|V_i|B_{ii} = \frac{Q_i - |V_i|^2 B_{ii}}{|V_i|}.
\end{aligned}$$

The matrices $\frac{\partial \mathbf{F}^P}{\partial \boldsymbol{\delta}}$ and $\frac{\partial \mathbf{F}^Q}{\partial \boldsymbol{\delta}}$ have the property that the diagonal element is the negative value of the sum of the other elements in the row. The Jacobian is a non-symmetric matrix.

Jacobian matrix gas network

The Jacobian of a gas network based on steady-state load flow (3.12) is given as

$$J_{Fg} = \begin{pmatrix} \frac{\partial \mathbf{F}^q}{\partial \mathbf{q}} & \frac{\partial \mathbf{F}^q}{\partial \mathbf{p}} \\ \frac{\partial \mathbf{F}^L}{\partial \mathbf{q}} & \frac{\partial \mathbf{F}^L}{\partial \mathbf{p}} \end{pmatrix}, \tag{5.5}$$

where conservation of mass system $\mathbf{F}^q$ is given by (3.14). If all links in the network represent gas pipes, then $\mathbf{F}_k^L$ is given by gas pipe link equation (3.15). The components of the Jacobian are given as

$$\begin{aligned}
\frac{\partial \mathbf{F}^q}{\partial \mathbf{q}} &= \mathbf{A}^{\mathbf{g}'},\\
\frac{\partial \mathbf{F}^q}{\partial \mathbf{p}} &= \mathbf{0},\\
\frac{\partial \mathbf{F}_k^L}{\partial q_k} &= -2(C^g)^{-2}|q_k|\left(f_k + \frac{1}{2}q_k\frac{\partial f_k}{\partial q_k}\right),\\
\frac{\partial \mathbf{F}_k^L}{\partial q_l} &= 0,\\
\frac{\partial \mathbf{F}_k^L}{\partial p_i} &= \begin{cases} 1, & \text{low-pressure,} \\ 2p_i, & \text{high-pressure.} \end{cases}\\
\frac{\partial \mathbf{F}_k^L}{\partial p_j} &= \begin{cases} -1, & \text{low-pressure,} \\ -2p_j, & \text{high-pressure,} \end{cases}
\end{aligned}$$

where link q_k represents gas flow from nodes v_i to v_j. It follows that $\partial \mathbf{F}^L/\partial \mathbf{q}$ is a diagonal matrix. If the pipe friction factor f_k is independent of the gas pipe flow, then $\partial f_k/\partial q_k = 0$. Note that in a low-pressure gas network we have

$$\frac{\partial \mathbf{F}^L}{\partial \mathbf{p}} = \mathbf{A}^{\mathbf{g}'^T},$$

the transpose of the reduced incidence matrix. The general matrix structure of the Jacobian of a low-pressure gas network where all links are gas pipes is

$$J_{F^g} = \begin{pmatrix} \mathbf{A}^{\mathbf{g}'} & \mathbf{0} \\ \operatorname{diag}\left[-2(C^g)^{-2}|\mathbf{q}|\left(\mathbf{f} + \frac{1}{2}\frac{\partial \mathbf{f}}{\partial \mathbf{q}}\mathbf{q}\right)\right] & \mathbf{A}^{\mathbf{g}'^T} \end{pmatrix}. \tag{5.6}$$

The Jacobian matrix of a gas network is thus a (non-symmetric) lower block-triangular matrix.

Jacobian matrix coupled energy network
The non-linear system of equations for steady-state load flow of a coupled energy network with electricity and gas is given by (4.7). The Jacobian of such a coupled electricity and gas network is given as

$$J_{F^c} = \begin{pmatrix} \frac{\partial \mathbf{F}^g(\mathbf{x}^g,\mathbf{x}^c)}{\partial \mathbf{x}^g} & \mathbf{0} & \frac{\partial \mathbf{F}^g(\mathbf{x}^g,\mathbf{x}^c)}{\partial \mathbf{x}^c} \\ \mathbf{0} & \frac{\partial \mathbf{F}^e(\mathbf{x}^e,\mathbf{x}^c)}{\partial \mathbf{x}^e} & \frac{\partial \mathbf{F}^e(\mathbf{x}^e,\mathbf{x}^c)}{\partial \mathbf{x}^c} \\ \mathbf{0} & \mathbf{0} & \frac{\partial \mathbf{F}^c(\mathbf{x}^c)}{\partial \mathbf{x}^c} \end{pmatrix} = \begin{pmatrix} J_{F^g} & \mathbf{0} & \frac{\partial \mathbf{F}^g(\mathbf{x}^g,\mathbf{x}^c)}{\partial \mathbf{x}^c} \\ \mathbf{0} & J_{F^e} & \frac{\partial \mathbf{F}^e(\mathbf{x}^e,\mathbf{x}^c)}{\partial \mathbf{x}^c} \\ \mathbf{0} & \mathbf{0} & \frac{\partial \mathbf{F}^c(\mathbf{x}^c)}{\partial \mathbf{x}^c} \end{pmatrix}. \tag{5.7}$$

The coupling parameters $\mathbf{x}_c$ consist of $\mathbf{q}_c$, $\mathbf{P}_c$ and $\mathbf{Q}_c$, as in (4.4). In the gas component $\mathbf{F}^g$, only the conservation of mass equations $\mathbf{F}^g_q$ can depend on $\mathbf{q}_c$. The matrix $\partial \mathbf{F}^g_q / \partial \mathbf{q}^c$ is thus a slice of the reduced incidence matrix of the coupled gas network of the columns corresponding to the dummy links. Denote this incidence matrix by A^q_c.

$$\frac{\partial \mathbf{F}^g}{\partial \mathbf{x}_c} = \begin{pmatrix} \frac{\partial \mathbf{F}^g_\mathbf{q}}{\partial \mathbf{q}_c} & \frac{\partial \mathbf{F}^g_\mathbf{q}}{\partial \mathbf{P}_c} & \frac{\partial \mathbf{F}^g_\mathbf{q}}{\partial \mathbf{Q}_c} \\ \frac{\partial \mathbf{F}^g_L}{\partial \mathbf{q}_c} & \frac{\partial \mathbf{F}^g_L}{\partial \mathbf{P}_c} & \frac{\partial \mathbf{F}^g_L}{\partial \mathbf{Q}_c} \end{pmatrix} = \begin{pmatrix} A^q_c & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{pmatrix}$$

A similar observation can be done for the energy network components. The coupling parameters $\mathbf{P}_c$ only occur in $\mathbf{F}^e_P$, and $\mathbf{Q}_c$ only in $\mathbf{F}^e_Q$. The matrices $\partial \mathbf{F}^e_P / \partial \mathbf{P}^c$ and $\partial \mathbf{F}^e_Q / \partial \mathbf{Q}^c$ are slices of the incidence matrix of the coupled electrical network of the columns corresponding to the dummy links. Denote these matrices by A^P_c and A^Q_c, respectively. Since a dummy link connects to only one node in a single-carrier network, the matrices A^q_c, A^P_c and A^Q_c are square with the number of rows/columns equal to the number of connected coupling units.

$$\frac{\partial \mathbf{F}^e}{\partial \mathbf{x}_c} = \begin{pmatrix} \frac{\partial \mathbf{F}^e_\mathbf{P}}{\partial \mathbf{q}_c} & \frac{\partial \mathbf{F}^e_\mathbf{P}}{\partial \mathbf{P}_c} & \frac{\partial \mathbf{F}^e_\mathbf{P}}{\partial \mathbf{Q}_c} \\ \frac{\partial \mathbf{F}^e_\mathbf{Q}}{\partial \mathbf{q}_c} & \frac{\partial \mathbf{F}^e_\mathbf{Q}}{\partial \mathbf{P}_c} & \frac{\partial \mathbf{F}^e_\mathbf{Q}}{\partial \mathbf{Q}_c} \end{pmatrix} = \begin{pmatrix} \mathbf{0} & A^P_c & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & A^Q_c \end{pmatrix}$$

The components of matrix $\frac{\partial \mathbf{F}^c}{\partial \mathbf{x}^c}$ depend on the equations used to model the coupling units.

The non-zero blocks in (5.7) are generally not square, due to the occurrence of new node types in coupled networks, see Section 4.2.2. Actually, if the Jacobian of a coupled network is upper block-triangular with square sub-matrices on the diagonal, the single-carrier parts of the network are not interdependent on each other [35].

5.1.3. Ill-Conditioned Jacobian Matrices

The convergence of Krylov solution methods is highly dependent on the conditioning of the coefficient matrix. If a matrix has poor spectral properties, the method will converge very slowly or even diverge. The application of an appropriate preconditioner should improve the condition number, and therefore the convergence behaviour of the Krylov solver. It is thus relevant to investigate how the coefficient matrices (Jacobians) are conditioned prior to applying an iterative solver. Literature mentions several aspects that worsen the conditioning of the Jacobian matrices of electricity and gas networks.

In [12], it was observed that electricity systems with more buses have larger condition numbers and that the eigenvalues take up larger regions than systems with less buses. The same observation was done for differently-sized gas networks in [40].

High and extreme loading of electricity systems leads to a higher condition number of the associated Jacobian [23, 48, 27]. High loading of an electricity system pushes it closer to an infeasible operation zone, leading to an ill-conditioned Jacobian. The methods used in [23] were able to solve the load-flow problems of sample networks with loading levels up to 160%, which they mention is beyond practical loading levels.

The electrical properties of the transmission lines also affect the condition number of the Jacobian of electricity systems [12, 27]. The R/X-ratio is the ratio of the inductive reactance and the resistance and a high ratio indicates that the system is more inductive. It results in the off-diagonal elements of the Jacobian to become much smaller than the diagonal elements, leading to near singularity.

A fourth aspect that influences the structure and conditioning of electricity system Jacobians is the topology of the network. In [27], it is mentioned that radial electricity systems have high condition numbers. In a radial electricity system, there is only one path in the network from a source to a load point. This topology occurs often in distribution systems. There are no loops in a radial network, thus a fault anywhere in the network will result in a loss of power for all downstream users. The corresponding Jacobian matrix has more zero-entries, leading to a highly sparse structure that can result in a high condition number. In [12], the condition numbers of longitudinal and meshed networks are compared. A longitudinal network models weak and long-distance power grids, often with a radial structure. A meshed network models an interconnected network where power can flow along multiple paths (loops). It was concluded that the longitudinal systems have condition numbers that are significantly higher than meshed systems. Additionally, they conclude that the location of the slack bus is more influential in meshed networks than in longitudinal networks.

5.2. Direct Solution Methods for Linear Systems

A commonly used direct solution method is *LU factorization*, which is based on the Gaussian elimination principle. In this method one factorizes the coefficient matrix A as

$$A = LU, \tag{5.8}$$

with L a lower-triangular matrix and U an upper-triangular matrix. This factorization is unique with the requirement that all diagonal elements of either L or U consist only of ones. The solution vector $\mathbf{x}$ is then found by sequentially solving the systems

$$L\mathbf{y} = \mathbf{f}, \quad U\mathbf{x} = \mathbf{y}.$$

The lower-triangular system with $L\mathbf{y} = \mathbf{f}$ is solved first using forward substitution. With the intermediate result $\mathbf{y}$, the upper-triangular system with $U\mathbf{y} = \mathbf{x}$ is solved using backward substitution, resulting in the solution vector $\mathbf{x}$.

Determining the solution vector $\mathbf{x}$ with LU factorization consists of computing the factorization (5.8), the forward substitution and backward substitution. For a dense coefficient matrix, the LU decomposition requires $\mathcal{O}(n^3)$ flops. The computational cost of both substitution steps is $\mathcal{O}(n^2)$ flops. The total complexity of determining the solution vector $\mathbf{x}$ is thus dominated by the factorization of the coefficient matrix.

LU factorization is used in solving load flow equations of single-carrier as well as multi-carrier networks. It is often used as the linear solution step in the non-linear NR solution method, as is mentioned in for example [47] and done in [39]. The NR method is discussed in more detail in 5.1.1.

Direct methods, such as LU decomposition, are often used in practice [48], as there are powerful direct solvers available that can solve large, sparse and ill-conditioned systems. Additionally, a direct method is usually less difficult to work with and implement than iterative methods. Methods based on LU decomposition have become the standard for the linear systems solve in the electricity systems community [47], and are also employed for small coupled gas-electric systems [37]. Current research on direct solvers for electricity systems focuses on their performance and implementation based on central processing unit (CPU)/graphics processing unit (GPU) accelerations [47, 10]. Direct solvers

scale poorly in problem size, increasing the computational efforts needed to converge significantly. This indicates that iterative solvers are more suitable to the problem.

5.3. Iterative Solvers for Linear Systems

In an iterative solution method, one starts with an initial guess $\mathbf{x}_0$ of the solution vector and aims to iteratively improve the approximation of $\mathbf{x}$. When after k iterations, the iterate $\mathbf{x}_k$ is sufficiently close to the exact solution $\mathbf{x}$, $\mathbf{x}_k$ can be used as an approximation. The *error vector* of the k-th iteration is defined as the difference between the exact solution and the solution approximation,

$$\mathbf{e}_k = \mathbf{x} - \mathbf{x}_k.$$

In practice, the exact solution $\mathbf{x}$ is unknown. Therefore, the *residual vector* is used instead. The residual vector of the k-th iteration is the difference between the right-hand side and left-hand side of (5.1),

$$\mathbf{r}_k = \mathbf{f} - A\mathbf{x}_k.$$

The residual can be computed for every solution approximation $\mathbf{x}_k$ and is used as a measure of the quality of the approximation. Note that the residual vector can be computed from the error vector via the residual equation $A\mathbf{e}_k = \mathbf{r}_k$.

There are various classes and types of iterative methods. The *basic iterative methods (BIMs)* such as Jacobi, Gauss-Seidel and Successive Overrelaxation are methods based on relaxation steps. In each iteration one or more components of the approximation are modified in a certain order such that components in the residual vector reduce to zero. These methods are rarely used on their own in practice, but form the basis of more advanced methods. Since they are not often used in solving load-flow equations, they are not discussed in more detail in this report. More about BIMs can be found in Chapter 4 of [49].

Other classes of methods employed in energy system modelling are *Krylov subspace* methods. These methods are commonly used in practice in many different fields of research and are discussed in Section 5.3.1. Section 5.3.2 discusses the generalized minimum residual (GMRES) method, which is popular due to its practical applicability. Another commonly used method is Bi-CGSTAB, the derivation of which is given in Appendix A. Finally, Section 5.3.3 presents an overview of Krylov methods used in energy systems modelling.

5.3.1. General Krylov Subspace Methods

The class of *Krylov subspace* methods is based on generating solution vector approximations $\mathbf{x}_k$ of the form

$$\mathbf{x}_k \in \mathbf{x}_0 + \mathcal{K}_k(A, \mathbf{r}_0), \tag{5.9}$$

where $\mathcal{K}_k(A, \mathbf{r}_0)$ is the Krylov subspace, defined as

$$\mathcal{K}_k(A, \mathbf{r}_0) = \text{span}\left\{\mathbf{r}_0, A\mathbf{r}_0, \dots, A^{k-1}\mathbf{r}_0\right\}. \tag{5.10}$$

Krylov subspace methods can be interpreted as projection methods. In a general projection method the k-th solution approximation is based on the initial guess $\mathbf{x}_0$, updated with a vector from the search subspace $\mathcal{K}_k$ such that the new residual has no components in the left subspace $\mathcal{L}_k$ (the new residual is orthogonal to all components in $\mathcal{L}_k$),

$$\begin{aligned} \mathbf{x}_k &= \mathbf{x}_0 + \boldsymbol{\delta}, \quad && \boldsymbol{\delta} \in \mathcal{K}_k, \\ (\mathbf{r}_0 - A\boldsymbol{\delta}, \boldsymbol{\omega}) &= 0, && \forall \boldsymbol{\omega} \in \mathcal{L}_k. \end{aligned}$$

In Krylov subspace methods the search subspace $\mathcal{K}_k$ is the Krylov subspace $\mathcal{K}_k(A, \mathbf{r}_0)$ as defined in (5.10). The choice of subspace $\mathcal{L}_k$ in part determines the type of Krylov subspace method. In orthogonal projection methods one chooses $\mathcal{L}_k = \mathcal{K}_k$, whereas in oblique methods one takes $\mathcal{L}_k = A\mathcal{K}_k$.

The general procedure of a Krylov subspace method is described in Algorithm 1. As can be seen from the general Krylov subspace method algorithm, there are different methods used to compute the

expanded basis vectors of the Krylov and left subspaces as well as computing the reduced systems. Two main methods to compute an orthogonal basis of the Krylov subspace are Arnoldi's orthogonalization method and Lanczos' biorthogonalization method. These methods give rise to different solvers.

Algorithm 1 General Krylov Subspace Method

1: Choose initial guess $\mathbf{x}_0$ and compute initial residual $\mathbf{r}_0 = \mathbf{f} - A\mathbf{x}_0$
2: **for** $k = 1, 2, \dots, n-1$ **do**
3: Expand the Krylov subspace $\mathcal{K}_k(A, \mathbf{r}_0) = \text{span}\{\mathbf{r}_0, A\mathbf{r}_0, \dots, A^{k-1}\mathbf{r}_0\}$
4: Generate/update basis vectors V_k of $\mathcal{K}_k(A, \mathbf{r}_0)$ and possibly W_k of $\mathcal{L}_k$ (method-dependent)
5: Form the projected matrix $H_k = W_k^T A V_k$
6: Solve the reduced system for $\mathbf{y}_k$ (method-dependent)
7: Update approximate solution $\mathbf{x}_k = \mathbf{x}_0 + V_k \mathbf{y}_k$
8: Compute residual $\mathbf{r}_k = \mathbf{f} - A\mathbf{x}_k$
9: **if** converged **then**
10: **break**
11: **end if**
12: **end for**

Arnoldi's orthogonalization method computes an orthonormal basis of a Krylov subspace $\mathcal{K}_{k+1}(A, \mathbf{r}_0)$. It determines the basis vector $\mathbf{v}_{k+1}$ that contains the components of $A^k\mathbf{v}_0$ not spanned by span $\{\mathbf{v}_0, \mathbf{v}_1, \dots, \mathbf{v}_{k-1}\} =$ span $\{\mathbf{v}_0, A\mathbf{v}_0, \dots, A^{k-1}\mathbf{v}_0\}$. This is done by multiplying previous basis vector $\mathbf{v}_k$ by A and orthonormalising it against all previous basis vectors by a standard Gram-Schmidt procedure. As a result, the basis V_k satisfies $V_k^T V_k = I$. The Arnoldi method results in the Arnoldi relation,

$$AV_k = V_k H_k + \mathbf{w}_k e_k^T, \quad V_k^T A V_k = H_k, \tag{5.11}$$

where H_k is the *Hessenberg* matrix. It represents how the coefficient matrix A acts inside the Krylov subspace. The shape of H_k is called upper-Hessenberg, where all entries below the first sub-diagonal are zero. It thus follows that solving a system with the Hessenberg matrix H_k is generally cheaper than solving a system with the original (possibly dense) coefficient matrix A.

The bi-Lanczos algorithm (see Appendix A) is cheaper to execute than the Arnoldi algorithm. In bi-Lanczos, basis vectors have to be orthogonalized against only the previous two, whereas for Arnoldi the new basis vector has to be orthogonalized against all previous basis vectors. Additionally, the matrix T_k that follows from bi-Lanczos requires less storage than the full Hessenberg matrix H_k. However, in the bi-Lanczos algorithm there are more opportunities for breakdown due to sensitivity to rounding errors.

In some scenarios, the coefficient matrix A is symmetric positive definite (SPD), for example when the Poisson equation is solved using a finite difference method. When the coefficient matrix is SPD, the Arnoldi method reduces to Lanczos. The SPD property is an advantageous property of the coefficient matrix to have and there are therefore several methods that make use of this property, such as the well-known conjugate gradient (CG) method. However, the coefficient matrices that arise from linearization of load flow problems of energy systems are generally not symmetric. This class of methods will thus not be discussed in this report, but are described in detail in [49].

5.3.2. The Generalized Minimum Residual (GMRES) Method

The *GMRES method* is a Krylov subspace method commonly used in practice for solving linear systems of equations [50]. Is it a projection method, taking $\mathcal{K} = \mathcal{K}_k$, the Krylov subspace, and $\mathcal{L} = A\mathcal{K}_k$ (oblique projection method). The iterations are based on Arnoldi's orthogonalization method. The procedure for computing the k-th iterate with GMRES is given in Algorithm 2. Each iteration of GMRES consists of the following steps:

1. Construct an orthonormal basis of the Krylov subspace $\mathcal{K}_k$ using Arnoldi's method by finding a vector $\mathbf{v}_k$ orthogonal to all vectors in V_{k-1}, an orthonormal basis of $\mathcal{K}_{k-1}$ (lines 3-14 in Algorithm 2).

2. Find the vector $\mathbf{y}_k$ that minimizes $||\mathbf{r}_k||_2$, the k-th residual (line 16 in Algorithm 2).
3. Compute iterate $\mathbf{x}_k = \mathbf{x}_0 + V_k\mathbf{y}_k$ (line 17 in Algorithm 2).

The second step of each GMRES iteration is to determine the vector $\mathbf{y}_k$ that minimizes the k-th residual. During this step, the Hessenberg matrix computed during Arnoldi's orthogonalization step is used to define a minimization problem that is equivalent to minimizing the residual, but is less costly to compute. Using the Hessenberg matrix and Arnoldi relation (5.11),

$$\begin{aligned} \mathbf{f} - A\mathbf{x} &= \mathbf{f} - A(\mathbf{x}_0 + V_k\mathbf{y}), \\ &= \mathbf{r}_0 - AV_k\mathbf{y}, \\ &= \beta\mathbf{v}_1 - V_{k+1}\bar{H}_k\mathbf{y}, \\ &= V_{k+1}\left(\beta\mathbf{e}_1 - \bar{H}_k\mathbf{y}\right). \end{aligned} \tag{5.12}$$

Since V_{k+1} is an orthonormal basis, it follows that computing the $\mathbf{x}_{k+1}$ that minimizes $||\mathbf{f} - A\mathbf{x}||_2$, is equivalent to finding the $\mathbf{y}_k$ that minimizes $||\beta\mathbf{e}_1 - \bar{H}_k\mathbf{y}||_2$ and computing $\mathbf{x}_{k+1} = \mathbf{x}_0 + V_{k+1}\mathbf{y}_{k+1}$.

The GMRES method is a suitable method for solving large power systems as it is a robust method for non-symmetric coefficient matrices A that is well-suited for preconditioning.

Algorithm 2 Generalized minimum residual method (GMRES) algorithm

1: Choose initial guess $\mathbf{x}_0$
2: $\mathbf{r}_0 = \mathbf{f} - A\mathbf{x}_0$, $\beta = ||\mathbf{r}_0||_2$, $\mathbf{v}_1 = \frac{\mathbf{r}_0}{\beta}$
3: **for** $j = 1, 2, \ldots, k$ **do**
4: $\quad \mathbf{w}_j = A\mathbf{v}_j$
5: $\quad$ **for** $i = 1, 2, \ldots, j$ **do**
6: $\quad\quad h_{ij} = (\mathbf{w}_j, \mathbf{v}_j)$
7: $\quad\quad \mathbf{w}_j = \mathbf{w}_j - h_{ij}\mathbf{v}_j$
8: $\quad$ **end for**
9: $\quad h_{j+1,i} = ||\mathbf{w}_j||_2$
10: $\quad$ **if** $h_{j+1,i} = 0$ **then**
11: $\quad\quad k = j$
12: $\quad$ **end if**
13: $\quad \mathbf{v}_{j+1} = \frac{\mathbf{w}_j}{h_{j+1,1}}$
14: **end for**
15: $\bar{H}_k = \{h_{ij}\}_{1 \le i \le k+1, 1 \le j \le k}$
16: $\mathbf{y}_k = \text{argmin}_{\ y}||\beta\mathbf{e}_1 - \bar{H}_k\mathbf{y}||_2$
17: $\mathbf{x}_k = \mathbf{x}_0 + V_k\mathbf{y}_k$

Restarted GMRES(m)
With the GMRES method as described above and in Algorithm 2, the number of vectors requiring storage increases with k, the dimension of the Krylov subspace. To remedy this difficulty, the GMRES algorithm can be restarted every m steps, where m is a fixed parameter. This restarted version is denoted by GMRES(m) [50]. After the restart, the current iterate $\mathbf{x}_m$ is taken as the initial guess for the next cycle of iterations [54]. In certain implementations, the GMRES(m) algorithm is often such that a restart either occurs after m iterations, or when the relative (preconditioned) residual decreased sufficiently or stagnates. The restarted GMRES algorithm can stagnate when the matrix A is not positive definite. The full GMRES algorithm is guaranteed to converge in at most n steps, where $A \in \mathbb{R}^{n\times n}$, but this is impractical when there are many iterations required for convergence.

5.3.3. Iterative Solution Methods in SC Energy Systems

Direct solvers scale poorly in problem size, increasing the computational efforts needed to converge significantly. This indicates that iterative solvers are more suitable to the problem. In research by Idema et al. [23], all considered preconditioned GMRES methods converged in less than one minute, whereas the direct linear solver took more than an hour to solve the large-scale linear problem. It is

thus generally recommended to use iterative solvers for solving large power flow problems [12, 38, 34]. The same results have been observed for transient gas networks [45]. In their experiments with large-scale networks, the direct solver either took up too much CPU time or ran out of memory, while the Krylov solvers converged. However, for smaller networks, the direct solver shows an advantage over the preconditioned Krylov solvers due to their big overhead. Thus, for smaller energy networks a direct solver is more suitable, while for larger networks the class of Krylov solvers should be considered.

The most commonly used Krylov solver for the linear solution step in power flow, as well as gas network simulations, is the GMRES method, although different arguments are given for this choice. The only general requirement for the coefficient matrix for GMRES is non-singularity, making it widely applicable, also to coupled networks [44]. Other argumentations for using GMRES is related to its behaviour and convergence properties. Full GMRES solves the problem in the least number of matrix-vector multiplications, which is the reason for Idema et al. [23] to use this method. Additionally, they state that the performance of GMRES can indicate whether bi-conjugate gradient stabilized (Bi-CGSTAB) or IDR(s) would be more suitable. Another advantage of GMRES is given in [48], where it is mentioned that GMRES is free of problems that interrupt the iterative process of constructing an orthonormal basis due to the underlying use of the Arnoldi method with Gram-Schmidt, opposed to methods based on Lanczos' method.

Another Krylov subspace method that is mentioned in literature is Bi-CGSTAB. The performance of Bi-CGSTAB is investigated in [38] as it is a variant of the extended (preconditioned) CG method, which had shown good convergence characteristics in DC power flow calculations. The preconditioned Bi-CGSTAB method outperformed the preconditioned conjugate gradient squared (CGS) method and preconditioned conjugate residual (CR) method in terms of computational effort. Additionally, preconditioned Bi-CGSTAB outperformed other Krylov methods such as CGS, GMRES and Bi-CG in the experiments done in [12]. In the application of steady-state gas networks, the authors of [40] observed similar performance for preconditioned GMRES, Bi-CGSTAB and IDR(s).

The iterative Krylov method applied to energy system modelling that is used most often in practice is GMRES. It is a robust method that is applicable to a large range of problems and serves as a good method to start with. Moreover, the choice of preconditioner is at least as important and influential on the convergence and performance as the selected method [49].

5.4. Domain Decomposition Methods

The class of *DD* methods consists of methods where a problem on a global domain is split into several problems on smaller domains, and iterating to determine the global solution. These methods revolve around the principle of 'divide-and-conquer', where a large problem is split into smaller sub-problems of the same type, until the sub-problems become simple enough to solve. Domain decomposition techniques can be advantageous, since sub-problems might be easier to solve, but also for their natural parallelizability. The DD principle additionally leads to a class of preconditioners that are based on the principle of divide-and-conquer. These are discussed in more detail in Section 6.2.3. This section describes the Schur complement approach and discusses applications of DD methods in single-carrier (SC) energy systems modelling.

5.4.1. Schur Complement Approaches

Assume that the linear system $A\mathbf{x} = \mathbf{f}$ can be written in block-form

$$\begin{bmatrix} B & E \\ F & C \end{bmatrix} \begin{bmatrix} \mathbf{y} \\ \mathbf{z} \end{bmatrix} = \begin{bmatrix} \mathbf{b} \\ \mathbf{g} \end{bmatrix}, \quad \text{with } A = \begin{bmatrix} B & E \\ F & C \end{bmatrix}, \ \mathbf{x} = \begin{bmatrix} \mathbf{y} \\ \mathbf{z} \end{bmatrix}, \ \mathbf{f} = \begin{bmatrix} \mathbf{b} \\ \mathbf{g} \end{bmatrix},$$

such that B is non-singular. The reduced system of $A\mathbf{x} = \mathbf{f}$ is given by

$$\left(C - FB^{-1}E\right)\mathbf{z} = \mathbf{g} - FB^{-1}\mathbf{b}. \tag{5.13}$$

The matrix

$$S = \left(C - FB^{-1}E\right),$$

is called the *Schur complement* of A. Solving the original system $A\mathbf{x} = \mathbf{f}$ is then equivalent to solving the reduced system, whose result can be used to compute

$$\mathbf{y} = B^{-1}\left(\mathbf{b} - E\mathbf{z}\right),$$

which leads to the solution vector $\mathbf{x}$.

Schur complement methods are based on solving the reduced system (5.13) by a (preconditioned) Krylov subspace method.

5.4.2. Domain Decomposition in Single-Carrier Energy Networks

Several applications of DD methods in SC electricity, gas and heat networks can be found in literature.

A study of Kootte, Sereeter, and Vuik [34] reviewed the numerical performance of connection methods of integrated transmission-distribution electricity networks. A transmission network is often modelled differently than a distribution network due to differences in assumptions, simplifications and physical properties. It is therefore not straightforward to integrate these separate domains. The study considers unified methods that solve the integrated network as a whole using a transformer, and splitting methods that keep the sub-networks separate to iteratively solve on the boundary of the domain. These splitting methods are also referred to as master-slave splitting (MSS) methods. The study results show that the unified methods outperform the splitting methods for the considered test cases, since splitting methods must solve the separate transmission and distribution networks multiple times until convergence, increasing the total CPU time. However, the authors mention that MSS methods are suitable in practical situations where full network information cannot be shared. For future work, they recommend to consider preconditioned Newton-Krylov methods for very large systems and note that MSS methods are parallelizable.

Zhang et al. [64] introduced a partial decoupling method for fast calculation of energy flow in large-scale heat and electricity integrated energy systems, that operate in a heat-load-following mode. They propose a partial decoupling method for district heating system (DHS) to decouple a network of arbitrary topology into several radial subsystems and one main system. Radial systems are preferred over looped systems as the hydraulic part of the load-flow equations of radial systems are linear. All unknowns of the subsystems are initialized, after which the DHS load-flow equations are solved iteratively. The values of the boundary nodes are updated after each subsystem solve. The global system has converged when the differences in temperatures and nodal mass flows of the boundary nodes in both systems become sufficiently small. This principle is similar to the multiplicative Schwarz (MS) procedure. Their partial decoupling method for various DHS performs about the same as traditional methods in terms of accuracy, while improving computational efficiency and sensitivity to improper hydraulic initialization.

The study of Srinivasan and Sundar [56] presents an algorithm that utilises a given partition of a network to compute a global solution for the full non-linear system through local solutions of smaller subsystems induced by the partition. Their research is on potential-driven steady-state non-linear network flow, but is applied and visualised on gas network datasets. They describe a bi-level reformulation of the load-flow equations, modifying the Jacobian matrix structure such that it is block-diagonal via a smart choice of interface nodes. These interface nodes are determined by considering permissible vertex separators, which are referred to as nodes whose removal increases the number of connected components, not connected to any other vertex separator. It is also shown that their method is equivalent to the Schur complement of the Newton step of the full non-linear problem. Their approach was tested successfully on gas network test datasets, but details on convergence and performance were not included. They emphasize that their method ensures that in practice, different network operators can solve their network flow system corresponding to their own domain, ensuring data sharing only occurs at the interface nodes.

6

Preconditioning

Preconditioning is a technique used to improve the efficiency and robustness of iterative solution methods. The principle is based on transforming the original linear system into a system that is easier to solve, but has the same solution vector. The quality of the preconditioner that is used is influential on the performance of the iterative method. Preconditioning is done by incorporating a non-singular preconditioning matrix M, such that the linear system $M\mathbf{x} = \mathbf{f}$ is cheap and easy to solve. This chapter first discusses how iterative solution methods can be preconditioned in Section 6.1. Next, Section 6.2 highlights different types of preconditioners by providing their purpose and derivations as well as their applications in energy systems modelling research.

6.1. Preconditioning Methods

The general principle of preconditioning is to transform the original linear system one aims to solve to an easier system that has the same solution. There are various ways to incorporate a preconditioner and thus to compute the preconditioned system. The three most common methods are *left*, *right* and *split* preconditioning.

A preconditioner can be applied from the left, resulting in the preconditioned system

$$M^{-1}A\mathbf{x} = M^{-1}\mathbf{f}. \tag{6.1}$$

Alternatively, the preconditioner can be applied to the right, leading to system

$$AM^{-1}\mathbf{u} = \mathbf{f},\ \mathbf{x} = M^{-1}\mathbf{u}. \tag{6.2}$$

When the preconditioning matrix can be factorized as $M = M_1M_2$, one can apply split preconditioning

$$M_1^{-1}AM_2^{-1}\mathbf{u} = M_1^{-1}\mathbf{f},\ \mathbf{x} = M_1^{-1}\mathbf{u}. \tag{6.3}$$

Note that taking $M_1^{-1} = I$ leads to right preconditioning as in (6.2), while $M_2^{-1} = I$ leads to left preconditioning as in (6.1).

Once a preconditioned system has been determined, it is solved using a linear solution method. The structure of the system matrix has an influence on the convergence behaviour of iterative linear solution methods, and most methods fit best to a certain types of matrix structures and properties. Since the choice of preconditioning matrix affects the preconditioned system matrix, the preconditioner has an effect on the convergence of the iterative solution method. The remainder of this section describes how the preconditioning matrix is incorporated in the generalized minimum residual (GMRES) Krylov methods.

6.1.1. Preconditioned GMRES

In the left-preconditioned GMRES algorithm, Arnoldi's orthogonalization method constructs an orthogonal basis of the left-preconditioned Krylov subspace

$$\mathcal{K}_k(M^{-1}A, M^{-1}\mathbf{r}_0) = \text{span}\left\{M^{-1}\mathbf{r}_0, M^{-1}AM^{-1}\mathbf{r}_0, \dots, (M^{-1}A)^{k-1}M^{-1}\mathbf{r}_0\right\}, \tag{6.4}$$

resulting from linear system (6.1). The algorithm of the left-preconditioned GMRES method is similar to the un-preconditioned GMRES as in Algorithm 2, but all actions of coefficient matrix A are replaced with $M^{-1}A$. A downside of the left-preconditioned GMRES method is that the un-preconditioned residuals, $\mathbf{r}_k$ are not explicitly computed. It is therefore difficult to define a stopping criterium based on these residuals, unless they are explicitly computed.

In the right-preconditioned GMRES algorithm, Arnoldi's orthogonalization method constructs an orthogonal basis of the right-preconditioned subspace

$$\mathcal{K}_k(AM^{-1}, \mathbf{r}_0) = \text{span}\left\{\mathbf{r}_0, AM^{-1}\mathbf{r}_0, \dots, (AM^{-1})^{k-1}\mathbf{r}_0\right\}, \tag{6.5}$$

based in linear system (6.2). The algorithm of the right-reconditioned GMRES method is similar to the un-preconditioned GMRES as in Algorithm 2, but all actions of coefficient matrix A are replaced with AM^{-1}, and the iterate solution computation is replaced with $\mathbf{x}_k = \mathbf{x}_0 + M^{-1}V_k\mathbf{y}_k$. The transformed variable $\mathbf{u}$ is not directly computed.

The preconditioned algorithms for GMRES can be found in Chapter 9 of [49]. The difference in convergence behaviour of the left- and right-preconditioned GMRES is not significant, unless the preconditioner M is ill-conditioned [49].

6.2. Preconditioning Techniques

Finding a good preconditioner is an important step in the application of iterative solvers, such as Krylov subspace methods. Despite general results and guidelines, there are no fixed or detailed theoretical results on how a preconditioner should be constructed and what their effect is on the solver performance. However, a general principle is that the preconditioning matrices are derived from the original coefficient matrix and the action of the preconditioner should be inexpensive to apply to a vector. This section includes the preconditioning techniques based on incomplete LU factorization and alternating Schwarz procedures.

6.2.1. Incomplete LU Factorization

An intuitive way of defining a preconditioner is by an *incomplete LU (ILU) factorization* of the original coefficient matrix,

$$A = \tilde{L}\tilde{U} - R,$$

where R is the residual matrix of the factorization, making the factorization incomplete. For a sparse matrix A, the matrices L and U often become more dense during the factorization process. The term *fill-in* refers to elements of the upper triangular part of U and lower triangular part of L that are non-zero, that were originally zero in the coefficient matrix A. This is an undesirable phenomenon since sparse matrices are less computationally expensive to work with.

There are different variations of ILU methods and preconditioners. Most ILU preconditioning methods aim at reducing the number of fill-ins so that the preconditioner is inexpensive to apply. How one determines which fill-ins to tackle differs per method and can be done in various ways. Most commonly either the location of the fill-in or its magnitude is used. Below, several commonly used ILU preconditioning techniques will be discussed. A more comprehensive list can be found in Section 10.3 of [49].

ILU(0)

The zero fill-in ILU factorization technique, denoted by ILU(0), takes the zero pattern of the incomplete factorization to be precisely the zero pattern of A. No fill-in is thus allowed, which is what the '0' indicates. The zero pattern of the factorization refers to the zero pattern of the lower triangular part of L and the

upper triangular part of U. Note that the product LU can therefore have a different zero pattern than A, see Figure 6.1.

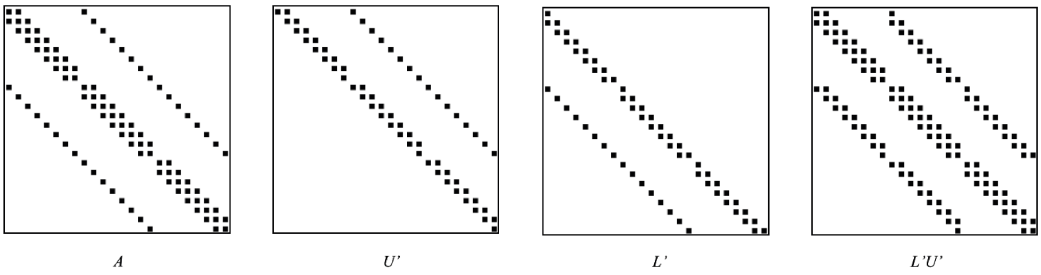


Figure 6.1: Taken from [38]: '*ILU(0) Factorization.*'

ILU(p)
In the ILU(p) factorization technique, all fill-in elements of level p or smaller are kept during the factorization process. For example, in ILU(1) only first-level fill-in elements are allowed. The rationale of the fill level is that the level of an element should be indicative of the size, such that a higher level corresponds to smaller elements. The initial level of fill of an element a_{ij} of sparse matrix A is defined by

$$\text{lev}(a_{ij}) = \begin{cases} 0, & \text{if } a_{ij} \neq 0 \text{ or } i = j, \\ \infty, & \text{otherwise.} \end{cases}$$

During the Gaussian elimination process (the LU factorization) the fill-level is updated through

$$\text{lev}(a_{ij}) = \min\left\{\text{lev}(a_{ij}),\ \text{lev}(a_{ik}) + \text{lev}(a_{kj}) + 1\right\}.$$

According to the above definition, the fill-level of an element does not increase during the elimination process. The zero pattern of ILU(p) may thus be written as

$$P_p = \left\{(i,j) : \text{lev}(a_{ij}) \leq p\right\}.$$

An example of the zero patterns of the ILU(1) factorization is given in Figure 6.2, based on the same coefficient matrix A as the zero pattern of the ILU(0) factorization from Figure 6.1.

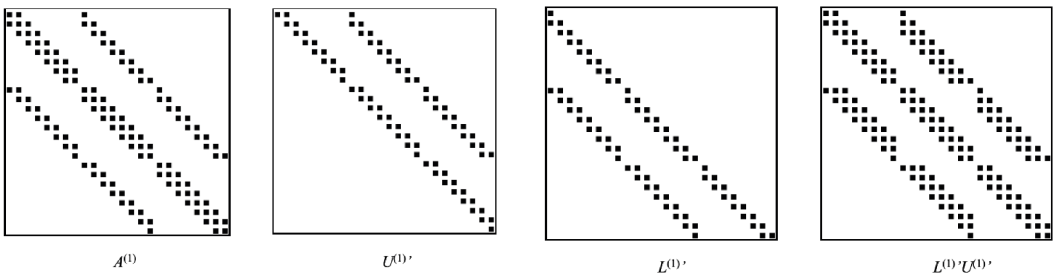


Figure 6.2: Taken from [38]: '*ILU(1) Factorization.*'

ILUT(p,τ)
Another commonly used ILU factorization uses a *dropping tolerance* in determining which fill-in elements to keep or discard. This can be done in various ways, but the general aim of applying a dropping rule to an element is to replace the element with a zero when it satisfies a set of criteria. In the ILUT approach, where 'T' stands for 'threshold', denoted by ILUT(p,τ), two general dropping criteria can be considered. First, during the Gaussian elimination process an element is dropped if it is less than the relative tolerance τ_i obtained by multiplying τ by the original norm of the i-th row. The goal of this criterion is to drop all elements that are of small magnitude. As the second dropping criteria, keep only the p largest elements in the row in addition to the diagonal element (which is always kept). The goal of this dropping rule is to control the number of elements per row, which can help reduce memory usage.

Pivoting
The ILUT approach can be combined with pivoting of the original coefficient matrix, leading to the ILUTP approach. Using pivoting is beneficial when the ILUT method encounters a zero pivot, over- or underflow or results in an unstable preconditioner. At the i-th step of the elimination process the largest entry of the row is selected to be the new i-th variable. The corresponding permutation matrices are updated accordingly.

Target matrix
An ILU preconditioner can be based on different matrices related to the coefficient matrix A. In the i-th Newton-Raphson (NR) iteration, the coefficient matrix is given by the Jacobian J_i. In [23], the performance of ILU preconditioners based on the Jacobian J_i, initial Jacobian J_0 and a Jacobian simplification known as *fast decoupled load flow (FDLF)* were compared. More about the conclusions of this research can be read in Section 6.2.2.

6.2.2. Incomplete LU Factorization in SC Energy Systems

The class of ILU preconditioners is often used in energy system modelling. Roque et al. [48] combined an ILU preconditioner with GMRES to study the performance of Krylov solvers for ill-conditioned systems resulting from the NR method applied to electricity systems. They investigate the effect *dropping tolerance* on the performance of the preconditioner. In the LU decomposition, every element with $|l_{ij}| < \epsilon_{ILU}$ and $|u_{ij}| < \epsilon_{ILU}$ is replaced with a zero, with dropping tolerances ϵ_{ILU} ranging from 10^{-3} to 10^{-7}. A larger tolerance drops more fill-in elements, making the preconditioner cheaper to apply, but the approximation of the ILU preconditioner of the original coefficient matrix is less accurate. As the dropping tolerance increases, the preconditioner gets closer to the full LU matrix and the linear system tends to converge faster. In later work, the authors mentioned that this tolerance-based dropping rule can introduce large errors, resulting in a low quality preconditioner [43]. As a remedy, they propose a dropping rule based on different criteria and highlight the effects reordering can have on the preconditioner quality.

Nguyen, Romate, and Vuik [40] also combined an ILU preconditioner with GMRES, but in the application of gas network load-flow equations. More specifically, the ILUTP preconditioner is used, which is based on an incomplete LU factorization with a dropping threshold and pivoting. They conclude that for the used test networks, the ILU preconditioner is not suitable, since the computation of the preconditioner requires nearly the same amount of work as for the LU factorization. In that scenario, using a direct solver based on LU is more suitable. For future research they recommend the Inexact-Newton method, keeping the same ILU factorization until the non-linear system has changed sufficiently.

Idema et al. [23] compared different ILU(k) preconditioners in simulations of large power flow problems, where good performance of ILU(k) with GMRES was observed. Additionally, they emphasize the importance of reordering methods in computing ILU factorizations. They concluded that using the approximate minimum degree (AMD) reordering for an ILU(k) preconditioner in GMRES resulted in around 25% less iterations needed for convergence compared to using no reordering. They also compared the performance of the ILU(k) preconditioner of different target matrices, the Jacobian J_i, the initial Jacobian J_0 and the FDLF matrix Φ. Using J_0 or Φ generally performed better than J_i. Since the J_0 preconditioner is the same in each iteration, only a single expensive factorization has to be made.

Mori and Iizuka [38] compared the performance of the ILU(k) preconditioner for $k = 0, 1, 2$ for three different Krylov subspace methods, applied to three differently sized electricity networks. For the largest test case considered, the use of the ILU(2) preconditioner resulted in the lowest total computation times as well as iterations. For the smaller test cases, the ILU(2) preconditioner always resulted in the fewest number of iterations needed for convergence, but not always the lowest computation time. This is attributed to the more expensive factorization of ILU(2) compared to ILU(1) or ILU(0).

The differences in performance and results of ILU preconditioned GMRES for electricity and gas networks, highlight that preconditioning is problem-dependent. The different versions and techniques used in ILU preconditioning such as reordering, dropping rule definitions and tolerances, all influence convergence and performance. On one hand this can lead to customized ILU preconditioners that

can enhance convergence of Krylov methods. On the other hand, this makes the search for the best combination of parameters a complicated procedure.

6.2.3. Schwarz Alternating Procedures

The methods referred to as Schwarz alternating procedures (SAPs), fall under a larger class of solution methods called *domain decomposition (DD)* methods. In these methods, a problem on a global domain is split into several problems on smaller domains. These subdomain solutions are then used to iteratively determine the global solution. These methods revolve around the principle of 'divide-and-conquer', where a large problem is split into smaller sub-problems of the same type, until the sub-problems become simple enough to solve. DD techniques can be advantageous, since sub-problems might be easier to solve, but also for their natural parallelizability. Saad [49] mentions that DD techniques are distinguished by four features: Type of partitioning, overlap between subdomains, processing of interface values, and subdomain solution methods and qualities. One method of processing interface values is demonstrated in the SAPs. These methods are based on alternating between overlapping subdomains, solving a new problem on each subdomain and exchanging boundary information. It will be shown that SAPs are equivalent to left-preconditioned fixed-point iterations.

Suppose we have subdomains Ω_i^δ such that

$$\Omega = \bigcup_{i=1}^{N} \Omega_i^\delta, \quad \Omega_i \subset \Omega_i^\delta,$$

with a δ-level overlap between neighbouring subdomains. There is a distinction between *multiplicative* and *additive* Schwarz procedures, based on how boundary information is exchanged. In the multiplicative Schwarz (MS) procedure, the algorithm sweeps through the N subdomains and solves the original problem in each subdomain by using the most recent boundary information (initial guess or solution of a previous subdomain). In the additive Schwarz (AS) procedure, the boundary components in each subdomain are updated after the sweep through all subdomains, not during. The MS procedure uses the most recent available information and will thus require less iterations than the AS procedure. However, the subdomain solves in the AS procedure can be done in parallel, which can reduce total computational time to find the global solution. Note that the MS procedure is analogous to the block Gauss-Seidel iteration, whereas the AS procedure is analogous to the block Jacobi iteration.

We aim to solve a general linear system $A\mathbf{x} = \mathbf{f}$, on global domain Ω using an iterative Schwarz procedure. Let $\Omega \subseteq \mathbb{R}^n$ and $\Omega_i^\delta \subseteq \mathbb{R}^{n_{i,\delta}}$ for overlapping subdomains Ω_i^δ. Define $R_i^\delta : \Omega \to \Omega_i^\delta$ as the *restriction operator* from global domain Ω to subdomain Ω_i^δ. It keeps only those components of a vector in Ω that are also in Ω_i^δ, and thus R_i^δ is an $n_{i,\delta} \times n$ matrix. Conversely, define the transpose of the restriction operator as a *prolongation operator*, ${R_i^\delta}^T : \Omega_i^\delta \to \Omega$, which takes the variables from a vector with components in Ω_i^δ and extends them to the equivalent components of Ω. The prolongation operator is thus an $n \times n_{i,\delta}$ matrix. Using the restriction and prolongation operators, we can define the associated linear system on a subdomain. Consider the subdomain system matrix

$$A_i = R_i^\delta A {R_i^\delta}^T. \tag{6.6}$$

Here, A_i is an $n_{i,\delta} \times n_{i,\delta}$ matrix, which is the restriction of global matrix A to subdomain Ω_i^δ.

Multiplicative Schwarz preconditioner

In the MS procedure, the global solution $\mathbf{x}$ is updated after each local solve. Here, the subscript $\cdot_i$ is used to denote the restriction of $\mathbf{x}$ to subdomain Ω_i, thus $\mathbf{x}_i = {R_i^\delta}^T \mathbf{x}$. The superscript $\cdot^i$ denotes the global solution $\mathbf{x}$ after updating with the solution of subdomain Ω_i^δ. With $\Delta\mathbf{x}_i$, denote the update on the subdomain Ω_i^δ, such that the global update is computed using the prolongation operator ${R_i^\delta}^T \Delta\mathbf{x}_i$. The local update is thus denoted by

$$\mathbf{x}_i^i = \mathbf{x}_i^{i-1} + \Delta\mathbf{x}_i, \tag{6.7}$$

while the global update is given by

$$\mathbf{x}^i = \mathbf{x}^{i-1} + {R_i^\delta}^T \Delta\mathbf{x}_i. \tag{6.8}$$

The local update is computed through solving local problem $A_i \mathbf{x}_i^i = \mathbf{f}_i$. Substitution of (6.7) in the local solve and rewriting results in

$$R_i^{\delta^T} \Delta \mathbf{x}_i = R_i^{\delta^T} A_i^{-1} R_i^{\delta} (\mathbf{f} - A\mathbf{x}^{i-1}).$$

This update is performed for each subdomain Ω_i^{δ}, and $\mathbf{x}_{\text{new}} = \mathbf{x}^N$ after all N subdomain updates. The MS procedure can be found in Algorithm 3. Note that in the update formula there is an inverse of the restricted coefficient matrix A_i. This does not necessarily imply that the inverse is directly computed, but indicates that the smaller subdomain problem has to be solved. Depending on the problem size and application, the resulting linear system subdomain problem can be solved with a direct or iterative method.

Algorithm 3 Multiplicative Schwarz procedure

1: **for** $i = 1, \dots, N$ **do**
2: $\quad \mathbf{x}^i = \mathbf{x}^{i-1} + R_i^{\delta^T} A_i^{-1} R_i^{\delta} (\mathbf{f} - A\mathbf{x})$
3: **end for**
4: $\mathbf{x}_{\text{new}} = \mathbf{x}^N$

From the MS procedure, one can derive that it is equivalent to solving a left-preconditioned system using a fixed-point iteration, which is done in the remainder of this paragraph.

The error after each subdomain update is given by $\mathbf{e}_i = \mathbf{x}^* - \mathbf{x}^i$, where $\mathbf{x}^*$ is the exact solution. The following relation holds for this error

$$\mathbf{e}_i = (I - P_i)\, \mathbf{e}_{i-1}, \quad P_i = R_i^{\delta^T} A_i^{-1} R_i^{\delta} A.$$

This relation indicates that the final error $\mathbf{e}_{\text{new}} = \mathbf{e}_N$ can be expressed using the P_i and initial error $\mathbf{e}_0$,

$$\mathbf{e}_{\text{new}} = \mathbf{e}_N = (I - P_N)\,(I - P_{N-1}) \dots (I - P_2)\,(I - P_1)\, \mathbf{e}_0.$$

Denote this product by Q_N such that

$$\mathbf{e}_{\text{new}} = Q_N \mathbf{e}_0, \quad Q_N = (I - P_N) \dots (I - P_1),$$

where the product in Q_N explains the name of the *multiplicative* Schwarz method. This relation implies that the updated solution vector $\mathbf{x}_{\text{new}}$ can be computed from $\mathbf{x}$ directly,

$$\mathbf{x}_{\text{new}} = Q_N \mathbf{x} + (I - Q_N)\mathbf{x}^*. \tag{6.9}$$

This MS procedure update formula is written as a fixed-point iteration. Let a general fixed point iteration be of the form

$$\mathbf{x}_{\text{new}} = G\mathbf{x} + \mathbf{b}. \tag{6.10}$$

For the MS update (6.9) we then have

$$G = Q_N, \quad \mathbf{b} = (I - Q_N)\mathbf{x}^*.$$

Note that a left-preconditioned system $M^{-1}A\mathbf{x} = M^{-1}\mathbf{f}$ is also solved using a fixed-point iteration of the form (6.10), where

$$G = I - M^{-1}A, \quad \mathbf{b} = M^{-1}\mathbf{f}. \tag{6.11}$$

Therefore, the MS procedure sweep is equivalent to a fixed-point iteration for the left-preconditioned system in which

$$M^{-1}A = I - Q_N, \quad M^{-1}\mathbf{f} = (I - Q_N)\mathbf{x}^*,$$

referred to as the *MS preconditioner*. The exact solution $\mathbf{x}^*$ is of course unknown and cannot be used directly in computing $\mathbf{x}_{\text{new}}$. However, one can operate with M^{-1} iteratively without having to explicitly form A^{-1}, using new definitions and recurrence relations. Define

$$T_i = R_i^{\delta^T} A_i^{-1} R_i^{\delta}, \quad Z_i = I - Q_i, \quad M_i = Z_i A^{-1}, \tag{6.12}$$

such that $M^{-1} = M_N$. Then the following recurrence relations hold

$$\begin{aligned} Z_1 &= P_1, \quad Z_i = Z_{i-1} + P_i(I - Z_{i-1}), \\ M_1 &= T_1, \quad M_i = M_{i-1} + T_i(I - AM_{i-1}). \end{aligned} \tag{6.13}$$

Using these definitions and recurrence relations, we can compute the action of $M^{-1} = M_N$ on an arbitrary vector $\mathbf{v}$ and on $A\mathbf{v}$ iteratively, see Algorithm 4. In the algorithm, the contribution of each subdomain solve to the preconditioner application is computed and added to form the final application of the preconditioner. Note that the T_i and P_i include local solves with A_i, which can be done using an appropriate direct or iterative method.

Algorithm 4 Multiplicative Schwarz preconditioner action on $\mathbf{v}$

1: Input: $\mathbf{v}$
2: $\mathbf{z}_1 = T_1\mathbf{v}$
3: **for** $i = 2, \dots, N$ **do**
4: $\quad \mathbf{z}_i = \mathbf{z}_{i-1} + T_i(\mathbf{v} - A\mathbf{z}_{i-1}) = \mathbf{z}_{i-1} + {R_i^\delta}^T A_i^{-1} R_i^\delta(\mathbf{v} - A\mathbf{z}_{i-1})$
5: **end for**
6: Output: $M_{MS}^{-1}\mathbf{v} = \mathbf{z}_N$

Additive Schwarz preconditioner
In the AS procedure, the updates from the local solves in the form of (6.8) are added to the global solution vector after all updates have been computed, leading to Algorithm 5.

Algorithm 5 Additive Schwarz procedure

1: **for** $i = 1, \dots, N$ **do**
2: $\quad \Delta\mathbf{x}_i = {R_i^\delta}^T A_i^{-1} R_i^\delta(\mathbf{b} - A\mathbf{x})$
3: **end for**
4: $\mathbf{x}_{\text{new}} = \mathbf{x} + \sum_{i=1}^N \Delta\mathbf{x}_i$

The update in line 4 of Algorithm 5 is already in a fixed-point form, (6.10), with

$$G = I - \sum_{i=1}^N P_i, \quad \mathbf{b} = \sum_{i=1}^N T_i A^{-1}\mathbf{f}.$$

The left-preconditioned system is solved using a fixed-point iteration with (6.11). Therefore, the AS procedure sweep is equivalent to a fixed-point iteration for the left-preconditioned system in which

$$M^{-1}A = \sum_{i=1}^N P_i, \quad M^{-1} = \sum_{i=1}^N T_i = \sum_{i=1}^N {R_i^\delta}^T A_i^{-1} R_i^\delta, \tag{6.14}$$

where the sum in M^{-1} explains the name of the *additive* Schwarz method. Using the same definitions (6.12) as for the MS preconditioner, the action of the preconditioner on an arbitrary vector $\mathbf{v}$ and $A\mathbf{v}$ can be computed iteratively, see Algorithm 6. Similarly to the multiplicative procedures, the contribution of each subdomain solve to the preconditioner application is computed and stored. Once all updates have been computed, the final application of the preconditioner is returned. Note that the computations in line 4 of Algorithm 6 can be done in parallel.

Schwarz Preconditioners and Krylov Subspace Methods
The multiplicative and additive Schwarz preconditioners can be applied when solving the general system $A\mathbf{x} = \mathbf{f}$ with a Krylov subspace method, such as GMRES. The algorithm of the preconditioned Krylov method is used, as described in Section 6.1, combined with the algorithms used to compute the

Algorithm 6 Additive Schwarz preconditioner action on $\mathbf{v}$

1: Input: $\mathbf{v}$
2: **for** $i = 1, \dots, N$ **do**
3: $\quad \mathbf{z}_i = T_i\mathbf{v} = {R_i^\delta}^T A_i^{-1} R_i^\delta \mathbf{v}$
4: **end for**
5: $\mathbf{z} = \mathbf{z}_1 + \cdots + \mathbf{z}_N$
6: Output: $M_{AS}^{-1}\mathbf{v} = \mathbf{z}$

action of the preconditioner on the desired vector, see Algorithms 4 and 6. These Schwarz preconditioners apply local corrections that are easier to fix than the global error. The overlap of the subdomains allows for propagation of the corrections such that the global error is smoothed. The combination of the local corrections results in a reduction of the global error. The level of overlap δ used in SAPs has an influence on the performance and effectiveness of the multiplicative and additive Schwarz preconditioners.

Additive Schwarz Variants: Interpolated and Restricted AS
Several variants of the AS preconditioner exist, that differ in the manner of using the local updates to construct the global solution. Two commonly used variants of the AS preconditioner are referred to as interpolated AS and restricted additive Schwarz (RAS).

The interpolated RAS preconditioner is a variant of the AS preconditioner where the updates of the overlapping indices are weighted. An index that is contained in more than one subdomain will receive an update from both subdomains, according to the AS procedure. This summation can result in over- or under-correction in these overlap areas. The interpolated AS preconditioner introduces a partition of unity (the weighting or interpolation operator) to combine local solutions. The interpolated AS preconditioner is defined as

$$M_{\text{IAS}}^{-1} = \sum_{i=1}^{N} {R_i^\delta}^T \mathbf{w}_i A_i^{-1} R_i^\delta, \tag{6.15}$$

where $\mathbf{w}_i$ contains the global weight of all elements in subdomain i. The weight is determined by the number of subdomains the element is contained in, after adding overlap. This way, the elements that are included in multiple subdomains receive an averaged update.

The RAS preconditioner is a variant of the AS preconditioner that reduces the number of iterations needed and CPU time required to compute the preconditioner, proposed by Cai and Sarkis [7]. The RAS preconditioner is defined as

$$M_{RAS}^{-1} = \sum {\hat{R}_i^0}^T A_i^{-1} R_i^\delta. \tag{6.16}$$

Here, ${\hat{R}_i^0}^T$ is the prolongation operator corresponding to a zero-level overlap ${R_i^0}^T$, extended with zero-columns such that it is a $n \times n_{i,\delta}$ matrix. Equivalently, ${\hat{R}_i^0}^T$ is the prolongation operator corresponding to a δ-level overlap, ${R_i^\delta}^T$, where the non-zero entries of the columns corresponding to the overlap points are set to zero. This RAS preconditioner uses known information from all points in Ω_i^δ, including overlap, to compute the local update. However, only the update for the internal points, so those in Ω_i^0, is used for the global update. As a result, half of the communication costs can be saved in a parallel implementation, since ${\hat{R}_i^0}^T A_i^{-1}$ does not involve any data exchange. Cai and Sarkis [7] found that the RAS preconditioner outperformed the standard AS preconditioner for all considered test cases, in terms of iteration counts, CPU time and communication costs if implemented in parallel. It was also shown that for M-matrices the RAS preconditioner does not require a damping parameter for convergence, while AS often does [18].

Two-level additive Schwarz preconditioners
The principle of a *multigrid* method can also be combined with AS preconditioners, referred to as *two-level* AS methods. In addition to the regular AS preconditioner, as in (6.14), that consists of the sum

of solves on subdomains, a coarse-level solve on a coarse domain is also done. This results in the two-level AS preconditioner, given by

$$M_{AS,2L}^{-1} = \sum_{i=1}^{N} {R_i^\delta}^T A_i^{-1} R_i^\delta + \Phi^T A'^{-1} \Phi, \tag{6.17}$$

where $\Phi : \Omega \to \Omega'$ is the restriction operator for coarse space Ω', and $A' = \Phi A \Phi^T$ is the restriction of the global matrix A to the coarse subdomain [58, 9]. The goal of the added coarse space correction to the preconditioner is to correct global errors that are not fixed by the local subdomain solves. The coarse subspace does not have to coincide or depend on the subdomains Ω_i^δ, but it is done often in practice. As an example, a node v_i in the coarse space representing a gas network can represent the average value of the nodal pressures for the nodes in subdomain Ω_i^δ.

6.2.4. Additive Schwarz Preconditioning in SC Energy Systems

The AS preconditioner is popular in research focussing on reducing total computation time to solve load flow problems, due to its natural parallelizability.

The overlapping RAS preconditioner applied to very large power flow problems was evaluated by Abhyankar, Smith, and Constantinescu [1] and in [2]. They combine the RAS preconditioner with the GMRES method to solve several large power flow problems. They compare the performance of the preconditioner with a block-Jacobi preconditioner and a parallel direct solver based on LU decomposition. Their results show a substantial speed-up of the RAS preconditioner compared to the other methods, ranging from 6 to 9 on 16 cores. The influence of the level of overlap was also investigated, and the authors concluded that a speed-up saturation and slowdown were observed for a level of overlap higher than 3. This observation is attributed to the increase in communications for higher levels of overlap.

7

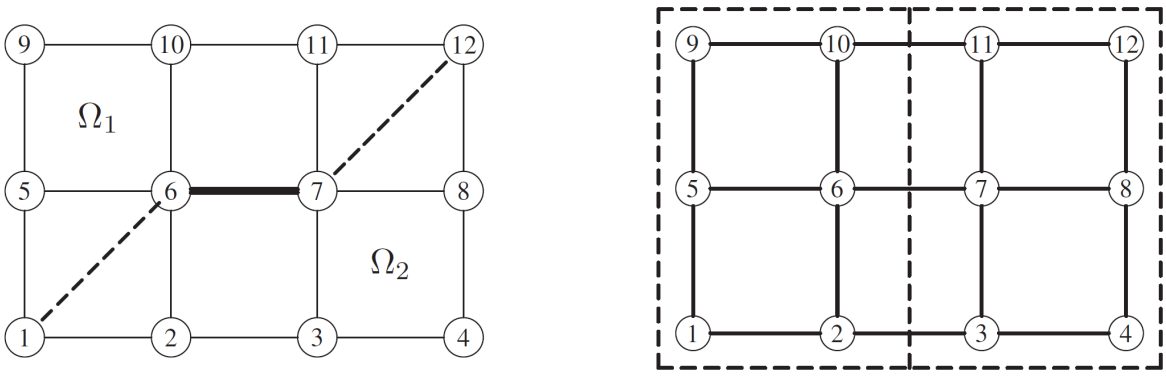
Partitioning of Energy Networks

The partitioning of single-carrier (SC) as well as multi-carrier energy networks (MCNs) occurs in various ways in simulations. For SC energy networks, a partition of the corresponding graph is made when using a domain decomposition (DD) method, or when applying a preconditioner based on Schwarz alternating procedures (SAPs). For MCNs, a partition of the graph is done often in decoupled methods to solve the MCN load flow equations based on the SC components. Therefore, it is necessary to have some understanding of various types of partitioning methods and their underlying principles. Section 7.1 introduces general graph partitioning methods and concepts, as well as multi-level partitioning algorithms often used in practice. Section 7.2 describes the usage of partitioning methods in energy network simulations.

7.1. Graph Partitioning

The method used for partitioning of an energy network directly determines the computational efficiency of preconditioner-based algorithms [8]. There are various methods to partition a graph, generally classified as *link-based* (or edge-based) and *node-based* (or vertex-based) partitioning, see Figure 7.1.

1. A link-based partitioning does not allow links to be split between two subdomains. Instead, the boundary is made up of vertices. Interface links are the links that connect nodes that do not belong to the same subdomain, such as the link connecting nodes 6 and 7 in Figure 7.1a.
2. A node-based partitioning does not allow vertices to be split between two subdomains such that the boundary consists of links. Interface nodes are the nodes that are connected by links belonging to at least two different subdomains, such as node 6 in Figure 7.1b. In a node-based partitioning, the links are cut to create the subdomains. The number of *link-cuts* refers to the number of links $e_k = \{v_i, v_j\}$ between nodes v_i and v_j, such that $v_i \in \Omega_\alpha$, while $v_2 \in \Omega_\beta$.



(a) Link-based partitioning

(b) Node-based partitioning

Figure 7.1: Taken from [49]: '*Link-based and node-based partitioning of a of a* 4×3 *mesh into two subregions.*'

A metric often used for node-based graph partitioning algorithms is the *edge-cut* (also called *cut-set*), which refers to the number of links (edges) whose incident nodes belong to different partitions [30]. In node-based partitions, the edge-cut gives an indication of how connected or separated the partitions are. When using a partition for an additive Schwarz (AS) or multiplicative Schwarz (MS) preconditioner, the edge-cut gives an indication of how much communication between the subdomains is required. An edge-cut is minimum if there is no other partition that results in a lower edge-cut. The sparsest cut problem refers to partitioning the nodes of a graph with the objective to minimize the ratio of the number of links across the cut, divided by the number of nodes in the smallest partition. This objective aims to both minimize the edge-cut while also favouring balanced partitions [8]. This sparsest cut problem is known to be NP-hard.

7.1.1. Multi-Level Partitioning Methods

A multi-level graph partitioning algorithm is an algorithm that determines a partition of a graph in different phases, generally the following three steps [28]:

1. **Coarsening phase**: The original graph is transformed into a sequence of coarsened graphs, such that each coarsening reduces the number of nodes.
2. **Partitioning phase**: A partitioning of the coarsened graph is determined, using a suitable partitioning algorithm.
3. **Un-coarsening and refinement phase**: The partition is projected back to the original graph and refined.

The steps are visualized in Figure 7.2. The aim is to reduce the computationally expensive problem of partitioning the original graph to a less expensive partition of a coarsened version of the graph. Reducing the graph to a coarser version is done with the aim of maintaining the graph structure as best as possible, while significantly reducing its complexity. During the second phase, the choice of partitioning method for the coarse graph should not only concern the quality of the coarse graph partition, but also the resulting partition of the original graph after the un-coarsening and refinement phase. During this last phase, local refinement heuristics should be used to improve the un-coarsened partitions.

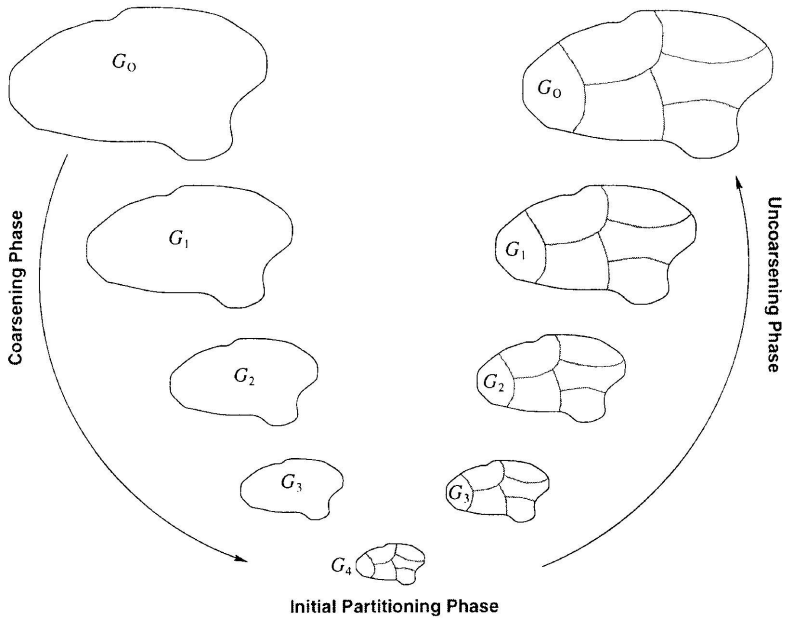


Figure 7.2: Taken from [30]: '*The various phases of the multilevel k-way partitioning algorithm. During the coarsening phase, the size of the graph is successively decreased; during the initial partitioning phase, a k-way partitioning of the smaller graph is computed (a 6-way partitioning in this example); and during the un-coarsening phase, the partitioning is successively refined as it is projected to the larger graphs.*'

Multi-level graph partitioning algorithms that are often applied in parallel processes are METIS, ParMETIS,

Scotch and CHACO [8, 21, 28, 29, 42]. The METIS algorithm is based on the three steps mentioned previously, and is widely applicable to both structured and unstructured graphs. It is an efficient algorithm for balanced partitions, but slower on irregular structures. The ParMETIS algorithm is a parallelized version of the METIS algorithm, using parallel methods in the coarsening and un-coarsening and refinement phase. Runtime improves with the parallelized algorithms, but scalability diminishes due to synchronization costs.

7.2. Partitioning Methods in SC Energy Networks

Various methods of partitioning SC and MCNs are used in research. Section 5.4.2 and Section 4.3.1 discussed the effects of the partitioning on solving the load flow equations of the various energy networks. In this section, the partitioning methods used in literature are mentioned and compared.

The coupled multi-energy system (MES) in the study of Markensteijn [35] are partitioned using a node-based partitioning, by cutting all dummy links that connect a homogeneous node to a coupling node. This cutting of dummy links results in two terminal links, one in each subdomain. Due to this natural partition, the subdomains consist of the individual electricity, gas and heat networks, as well as a subdomain with only coupling nodes. A similar partitioning was used in [64] to decouple an electro-thermal system into several district heating systems (DHSs), one large electricity system and coupling units.

The partial decoupling method for a DHS used by Zhang et al. [64] makes use of link-based partitioning (see Section 7.1) by splitting an intermediate node (branch bus, no load or source) into a load node for the first subsystem, and a slack node for the second subsystem. The goal of the partitioning is to create only radial subsystems that are more robust regarding initialization.

Abhyankar, Smith, and Constantinescu [1] also make use of a partitioning method of electricity systems in order to use a restricted additive Schwarz (RAS) preconditioner. It is mentioned that obtaining a good graph partition is a critical aspect for parallel linear solution methods, but that it is beyond the scope of their research. For the partitioning of the electricity network, they used a multi-level graph partitioning algorithm.

The study of [8] compares efficient graph partitioning algorithms for sparse matrices, considering various multi-level algorithms. The application of METIS and ParMETIS, among other methods, to standard datasets showed that METIS had consistent performance, ideal for small and medium number of partitions. ParMETIS demonstrated superior scalability for a higher number of partitions.

From the literature mentioned above we can see that partitioning used in energy systems can be based on the physical structure of the network as well as computational optimality. The physics-based partitioning used to partition a MCN is generally node-based, such that nodes of the same carrier are partitioned together. The partitioning of nodes of the same carrier is done in various ways, either focussing on accuracy of the obtained solution of the partitioned system or computational efficiency.

8

Methodology and Experimental Setup

The aim of this research is to gain insights into the performance of the additive Schwarz (AS) preconditioner when applied in the solution process of the load flow analysis of energy networks. These energy networks can be of single-carrier (SC) type, such as gas and electricity networks, or be a coupled energy network. The application of the AS preconditioner has been studied in the context of electricity networks [1], showing good performance for large-scale networks. For gas networks a domain decomposition (DD) method has been applied [56], but AS preconditioning in gas networks has not yet been considered. For coupled energy networks, various studies on integrated and decomposed methods were done [26, 35, 44, 64]. These studies show promising results for smaller networks and suggest further research into DD type methods for coupled load flow analysis.

This chapter establishes the methodology and experimental framework used to evaluate the performance of the AS preconditioner, applied in the solution process of steady-state load flow analysis of gas, electricity and coupled networks. Section 8.1 describes the datasets used to which the AS preconditioner is applied. Next, Section 8.2 discusses the implementation of the AS preconditioner used in this research, including choices made during the implementation phase. The last section, Section 8.3 includes the set-up of the conducted experiments and the performance metrics used to analyse and quantify the AS preconditioner performance.

8.1. Test Networks and Data

This section introduces the datasets used in this study to analyse the AS preconditioner performance. For gas networks, the GasLib datasets [53] are considered, as was also done in the research into steady-state gas load flow by Nguyen, Romate, and Vuik [40] and Srinivasan and Sundar [56]. For electricity networks, the datasets included in the PandaPower database are used [57], which includes the IEEE datasets often used in research into steady-state load flow of electricity networks. The largest PandaPower datasets were also considered in the research by Abhyankar, Smith, and Constantinescu [1], who applied the AS preconditioner to large power flow problems. This section concludes by describing the construction of coupled energy network data used in this research.

8.1.1. Gas Networks: GasLib

The test networks and data for the experiments with gas networks in this research are taken from the GasLib database [53]. It is an open-source database with data of gas transmission networks based on the German gas network. The gas networks used from the database are GasLib-11, GasLib-24, GasLib-40, GasLib-135, GasLib-582, GasLib-2607 and GasLib-4197, where the number in the name indicates the number of nodes in the network. A summary of the GasLib gas networks is given in Table 8.1. A selection of the corresponding graph representations is given in Appendix B, in Section B.1. The nodes in the GasLib networks are either sources, sinks or junctions. A link can represent a gas pipe, compressor, resistor, short pipe, valve or control valve. The gas type used in this research is hydrogen gas. Alternatively, natural gas can be used.

The network GasLib-11, the smallest available GasLib network, is used in this research to illustrate the partitioning methods and the effect of the AS preconditioner on the network Jacobian. The networks GasLib-11, GasLib-24, GasLib-40 and GasLib-135 are considered small-scale networks. The GasLib-582, GasLib-2607 and GasLib-4197 datasets are used as large-scale test networks.

Table 8.1: Summary of the gas network test data from GasLib [53], including the size classification within this research.

Network name	Nodes	Links	Jacobian size	Classification
GasLib-11	11	11	21×21	Small
GasLib-24	24	25	48×48	Small
GasLib-40	40	45	84×84	Small
GasLib-135	135	170	304×304	Small
GasLib-582	582	609	1190×1190	Large
GasLib-2607	2607	3083	5689×5689	Large
GasLib-4197	4197	4466	8662×8662	Large

8.1.2. Electricity Networks: PandaPower

The test networks and data for the experiments with electricity networks are taken from the PandaPower database [57]. It is an open-source `Python` library that combines the data analysis library `Pandas` and the power flow solver `PYPOWER`. In this research, only the network datasets of PandaPower are used, that consist of the data from the `PYPOWER` library and `MatLAB MATPOWER` toolbox. The test networks range from a 4-node network to a 9241-node network. The number in the name indicates the number of nodes in the network. The datasets are based on various European and American transmission networks. A summary of the PandaPower electricity networks is given in Table 8.2. A selection of the corresponding graph representations is given in Appendix B, in Section B.2.

This research considers a selection of the test networks in the PandaPower database. The network case9, consisting of 9 nodes, is used as the smallest network to illustrate the partitioning methods and the effect of the AS preconditioner on the network Jacobian. The networks case30 and case89pegase are used as small-scale test networks to investigate the performance of the AS preconditioner.

Table 8.2: Electricity network test data from PandaPower [57], with the network name, information on the types of nodes and links, the size of the corresponding Jacobian matrix and the size classification.

Network name	Nodes	Generators	Loads	Lines	Jacobian size	Classification
case9	9	2	3	9	14×14	Small
case14	14	4	11	15	22×22	Small
case30	30	5	20	41	53×53	Small
case89pegase	89	17	29	160	165×165	Small
case300	300	79	193	283	530×530	Small
case2869pegase	2869	689	1311	4051	5227×5227	Large
case6495rte	6495	1644	3380	7416	12491×12491	Large
case6515rte	6515	1650	3410	7422	12536×12536	Large
case9241pegase	9241	1878	4461	13797	17036×17036	Large

8.1.3. Coupled Networks

The test networks and data used for the experiments with coupled networks are constructed using the PandaPower database for the electrical network component, and the GasLib database for the gas network component. For this research, a coupled network consists of one gas network component and one electrical network component, with one or more coupling units between the two components.

Due to restrictions on nodes suitable for coupling power-to-gas (P2G) and gas-fired generator (GFG) units, it is not possible to couple all PandaPower electrical networks to all GasLib gas networks using an arbitrary number of coupling units. The restrictions are further elaborated on in Section 8.2.2.

Appendix C contains Table C.1 and Table C.2, which indicate the maximum number of P2G and GFG coupling units possible per SC network.

Within a coupled energy network, there can be one-sided or two-sided interaction. With one-sided interaction, the steady-state solution can be found by first computing the solution to one SC component, which automatically determines the steady-state solution of the second SC component through the coupling unit(s). The SC that determines the input or output for the coupling unit, is referred to as the *leading carrier*. When all node type changes occur in the leading carrier, there is one-sided interaction. Note that when one coupling unit is present, the interaction is always one-sided. Two-sided interaction occurs when the conversion of energy can go both ways, thus when both networks have adjusted node types. The test networks used in this research consist of both one-sided interaction with P2G and GFG units, as well as two-sided interaction scenarios.

Since it is insightful to also investigate the effect the coupling units have on the AS preconditioner and solution method, it was decided to consider the PandaPower and GasLib networks that allow both P2G and GFG coupling units. From Table C.2, one can see that for most small electrical networks there are no nodes suitable for a GFG unit. Only the case89pegase and case300 datasets are suitable for coupling both P2G and GFG units, where the total number of nodes in the dataset is less than 1000. Therefore, these datasets will be used as the electrical network components within the smaller coupled networks. All GasLib datasets are suitable for both P2G and GFG unit coupling. Although coupling using a GFG generator is not possible for the smallest PandaPower network, the case14 dataset coupled to the GasLib-11 network using a P2G unit is used to more clearly illustrate the coupling configuration and Jacobian matrix structure. Table 8.3 gives an overview of coupled networks used as test data in this research.

Table 8.3: Coupled networks test data, indicating the SC network components and possible coupling interactions.

Electrical Network	Gas Network	Interaction
case14	GasLib-11	One-sided
case89pegase	GasLib-40	One-sided and two-sided
case2869pegase	GasLib-2607	One-sided and two-sided
case9241pegase	GasLib-4197	One-sided and two-sided

8.2. Implementation

For this research, the AS preconditioner is implemented within an existing codebase. With this software, simulations of steady-state gas, electricity and coupled energy networks can be performed and analysed. The same codebase was used in [40]. This section includes all information related to the implementation of the AS preconditioner, as investigated in this research. This includes the general solver framework of the employed codebase, the construction of coupled networks, the partitioning and translation of network graphs to the linearised systems and the implementation of the AS preconditioner variants. The software used in this research is available on GitHub [61].

8.2.1. Solver Framework

The codebase is implemented in `Python` in an object-oriented manner. Electricity and gas network data are extracted and stored in `Pandas` data frames. It makes use of the standard `NumPy` library to perform numerical linear algebra operations, where sparse matrices are stored and handled in sparse formats using the `SciPy` library. Visualisations of data and results are generated with the `Matplotlib` library. In order to monitor iteration counts and times, the `tqdm` library was also utilized.

Additionally, `PETSc` [4] is used in the codebase via `petsc4py` [11], the `Python` wrapper for `PETSc` libraries. `PETSc` (the portable, extensible toolkit for scientific computation) is a scalable (parallel) solution of scientific application modelled by partial differential equations, primarily written in `C`. It is intended for use in large-scale application projects and includes an expansive suite of parallel linear and nonlinear equation solvers.

The non-linear systems corresponding to the steady-state load flow calculations of SC energy networks are computed and constructed in the codebase using the methods and equations as described in Chapter 2 and Chapter 3. The coupled networks and their load flow equations are constructed using the principles in Chapter 4. Solving the non-linear system of equations is done using the Newton-Raphson (NR) method, see Section 5.1. During each non-linear iteration, a (preconditioned) linear system is solved. The implemented solvers are LU decomposition (direct linear solve), and the iterative solvers generalized minimum residual (GMRES), bi-conjugate gradient stabilized (Bi-CGSTAB), IDR and least-squares QR (LSQR). All solvers, except IDR, are the corresponding solvers of the `SciPy` library. The included preconditioners are Jacobi, Gauss, incomplete LU (ILU) and Kaczmarz. Alternatively, the solvers of the `PETSc` library can be used. If the linear solver is chosen to be one of the `PETSc` solvers, any of the `PETSc` preconditioners can be used.

8.2.2. Coupled Networks

Within the codebase, a heterogeneous coupled network is created by first creating two separate networks, one with gas and one with electricity as a carrier. The type of coupling unit is determined beforehand, as well as the number of each type of coupling unit. Chapter 4 discusses the coupling and coupling units in more detail. Next, a heterogeneous network is created that consists of the coupling nodes. The coupling units available in the codebase for coupled gas-electricity networks are P2G and GFG.

Power-to-gas (P2G)

A P2G unit converts energy from electricity to gas. As a result, in a P2G unit energy can only flow from the electrical node to the gas node. In the codebase, a P2G unit can only be coupled to an electrical PQ node, where the real power is positive. Such a PQ node corresponds to an electrical network element where energy is already leaving the electrical network. Similarly, the gas node that the P2G unit is connected to has to be a load node with negative gas mass flow. For such a load node, there is already energy entering the gas network. In theory, a P2G unit can be connected to any electrical and gas node. It was decided to only connect P2G units to the specific nodes described above in the codebase, due to a more realistic interpretation and implementation of real-life coupled networks and to simplify the implementation in the codebase.

The coupling variables associated with a P2G unit are the dummy link variables. A gas dummy link has gas mass flow associated with it, denoted with q_c. The electrical dummy link results in a real coupling power, denoted with P_c. The P2G unit introduces one coupling equation, see (4.2). In order for the linearized coupled system to be square, at least one more system variable has to be considered known. Within the codebase, the node type of the electrical node the unit is connected to is changed from a PQ node to a PQV node. The voltage magnitude at this node is set to its reference voltage. Beside this node type change, it is also possible to change any other node type (see Section 4.2.2). The codebase uses this type of node change due to the interpretation and feasibility for a real-life coupling unit and to simplify the implementation in the codebase.

Gas-fired generator (GFG)

A GFG unit converts energy from gas to electricity. As a result, in a GFG unit energy can only flow from the gas node to the electrical node. Within the codebase, a GFG unit can only be coupled to a PQ electrical node, where the injected real power is negative. This corresponds to a load node where energy is already entering the electrical network. The gas node that the GFG is connected to must be a load node with positive gas mass flow, such that energy is already leaving the system at that node. Similarly to a P2G node, a GFG unit can be connected to any electrical and gas nodes in theory. It was decided to only connect GFG units to the specific nodes described above in the codebase, due to a more realistic interpretation and implementation of real-life coupled networks and to simplify the implementation in the codebase.

The coupling variables associated with a GFG unit are the same as for a P2G unit, the dummy link variables for gas mass flow and real power. The GFG unit introduces one coupling equation, see (4.1). In order for the linearized coupled system to be square, at least one more system variable has to be considered known. Within the codebase, the node type of the gas node the unit is connected to is

changed from a load node to a reference load node, where both the nodal pressure p and injected gas mass flow q are known. The pressure at this node is set the same as the pressure on the slack node. Beside this node type change, it is also possible to change any other node type (see Section 4.2.2). The codebase uses this type of node change due to the interpretation and feasibility for a real-life coupling unit and to simplify the implementation in the codebase.

8.2.3. Graph Partitioning

The goal of partitioning an energy network is to determine the sub-systems for the AS preconditioner. This is done by creating a partition of the energy network graph, and translating the partition to construct sub-matrices of the global Jacobian matrix.

Partitioning the Network Graph: Single-Carrier Networks

A partition of the nodes of an energy network is made using built-in partitioning functions of the `PETSc` library. The partitioning algorithms that one can use are ParMETIS, METIS, CHACO and average. These multi-level partitioning algorithms are described in Chapter 6. The average partitioning algorithm creates a partition based on the node index in ascending order, and thus does not take the graph structure into account. The ParMETIS, METIS and CHACO partitions are constructed using the graph adjacency matrix. Electricity networks consist of undirected graphs, and as a result, the non-zero structure of the network admittance matrix can be used as the adjacency matrix. The adjacency matrix of a gas network is constructed in order to create a partition using the mentioned multi-level algorithms. For all partitioning algorithms, the number of subdomains has to be specified beforehand. The node types of the SC networks are not taken into account when constructing the partition. The subdomains are a result of a purely graph-based partitioning algorithm.

Partitioning the Network Graph: Coupled Networks

For this research, it was decided to partition a coupled energy network graph per carrier. In this way each subdomain has one SC network as main network. Since the coupled network always consists of an electricity and gas network, there are always 2 subdomains. The first subdomain contains all gas nodes and links, as well as the gas dummy links. The second subdomain consists of all electrical nodes and the electrical dummy links.

Other methods of partitioning a coupled energy network are also possible. For example, a partition per carrier can be made, as described above, after which the gas and electrical network components are again divided into multiple subdomains. Alternatively, a partition can be made based on the entire coupled network graph, where the carrier of each node is not (necessarily) taken into account. In this research the carrier-based partition is made to study the effect of coupling units and interaction between different carriers, and to simplify the implementation in the codebase. The usage of other coupled network partitioning methods is included in the discussion in Section 11.3.

Including the Links

The load flow equations of an electricity network consist only of nodal equations. It is thus not necessary to include the links into a partition.

The load flow equations of a gas network consist of nodal as well as link equations. Therefore, the links should also be included in the partition. There are various methods of including the links. Two link-inclusion methods are considered in this research.

- A simple link-inclusion method is to include those links in subdomain i that are outgoing links of the nodes in subdomain i. As a result, each link is included in exactly one subdomain.
- A second link-inclusion method is to include those links in subdomain i that are adjacent to the nodes in subdomain i. As a result, internal links are included in exactly one subdomain, while links between nodes in different subdomains are included in both the subdomains.

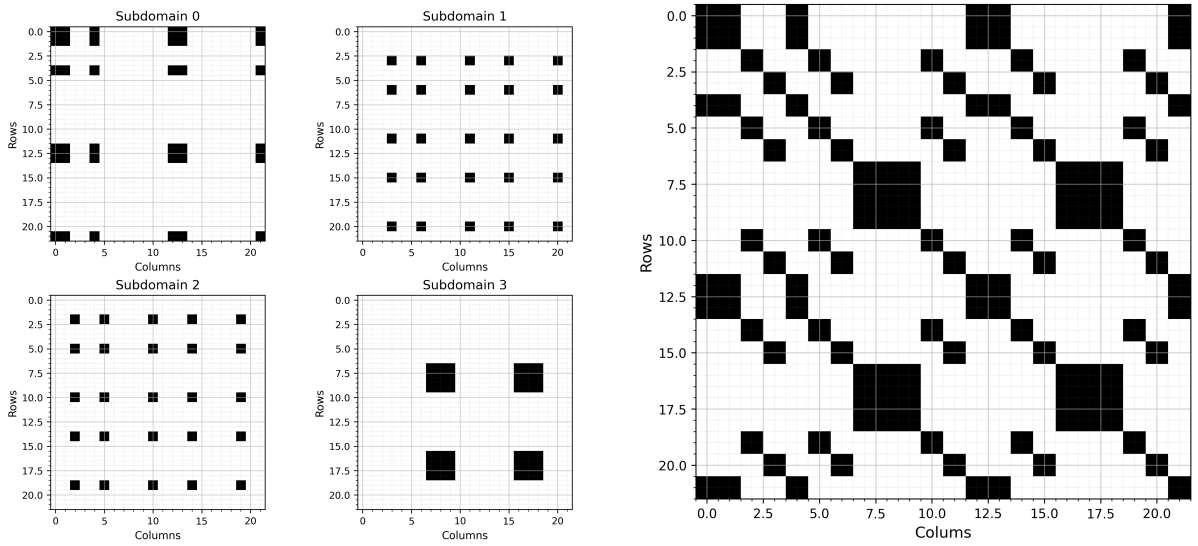
When the simple link-inclusion method is used such that only outgoing links are included in the subdomain, it is not guaranteed that the corresponding sub-system Jacobian is non-singular. When a subdomain graph consists of n nodes, the minimum number of links is $n-1$, a minimal spanning tree (under the assumption that the graph is connected). It was found during implementation that when

there are more nodes than links in a subdomain, the corresponding sub-system Jacobian matrix has two linearly dependent rows in the upper left $\partial \mathbf{F}^q / \partial \mathbf{q}$ block, see Equation 5.6. The $\partial \mathbf{F}^q / \partial \mathbf{q}$ block is given by the reduced incidence matrix. The incidence matrix of a graph with more nodes (rows) than links (columns), always has linearly dependent rows. When the sub-system Jacobian matrix is singular, the construction of the AS preconditioner fails. A subdomain must thus contain at least as many links as nodes. Therefore, the simple link-inclusion method is not an appropriate in the setting of AS preconditioning.

When the second link-inclusion method is used such that all incoming and outgoing links are included in the subdomain, it is guaranteed that there are at least as many links as nodes in the subdomain. When a subdomain with n nodes is a minimal spanning tree, there are $n - 1$ internal links. Since the global graph is connected, there is at least one link in the subdomain that is not internal, but is connected to a node in another subdomain. The minimum number of links included in the subdomain is therefore n. Thus, with the second link-inclusion method, it is guaranteed that a subdomain contains at least as many links as nodes. This research will therefore only consider the second link-inclusion method.

Overlap
An overlap is added to each subdomain by including nodes adjacent to the internal nodes of the subdomain. A δ-level overlap refers to including all nodes that are δ steps away from any internal node, where $\delta \in \mathbb{N}$. Section 6.2.3 describes the δ-overlap notation and principle in more detail. For a gas network graph, the links adjacent to the overlap nodes are also included, according to the employed link-inclusion method.

Translating to a Partition of the Jacobian: Electricity Networks
The partition of the electricity network graph has to be translated to a partition of the corresponding linearized system. Each node in the graph results in either 0, 1 or 2 equations in $\mathbf{F}^e$ and unknowns in $\mathbf{x}^e$, depending on the node type. The node types of electricity networks are described in Section 3.1.2. A slack node does not directly correspond to an equation or unknown, due to the combination of known and unknown nodal parameters. Similarly, a PV node (generator) corresponds to one equation and one unknown, while a PQ node (load) results in two equations and unknowns. The partition of the network graph is translated to the linearized system by taking those elements of the global Jacobian which correspond to the equations and unknowns associated with the nodes in that subdomain. Thus, global Jacobian element $\partial \mathbf{F}_i^e / \partial \mathbf{x}_j^e$ is only included in the sub-system Jacobian matrix when the node corresponding to equation $\mathbf{F}_i^e$ as well as the node corresponding to unknown x_j^e are part of the corresponding subdomain. An example of the partitioning of the global Jacobian is given in Figure 8.1. The node types are not taken into consideration when constructing the partition. As a result, when a subdomain consists of n nodes, the size of the sub-system Jacobian matrix can range between $(n-1)\times(n-1)$ and $2n \times 2n$ (assuming that a subdomain contains at most one slack node).



(a) Entries of the global Jacobian included in each subdomain Jacobian.

(b) Entries of the global Jacobian included in a subdomain.

Figure 8.1: Subdomain inclusion pattern of the global Jacobian, indicating which entries are included in at least one subdomain as a result of partitioning the PandaPower case14 graph into 4 subdomains using the `PETSc` ParMETIS algorithm, without overlap.

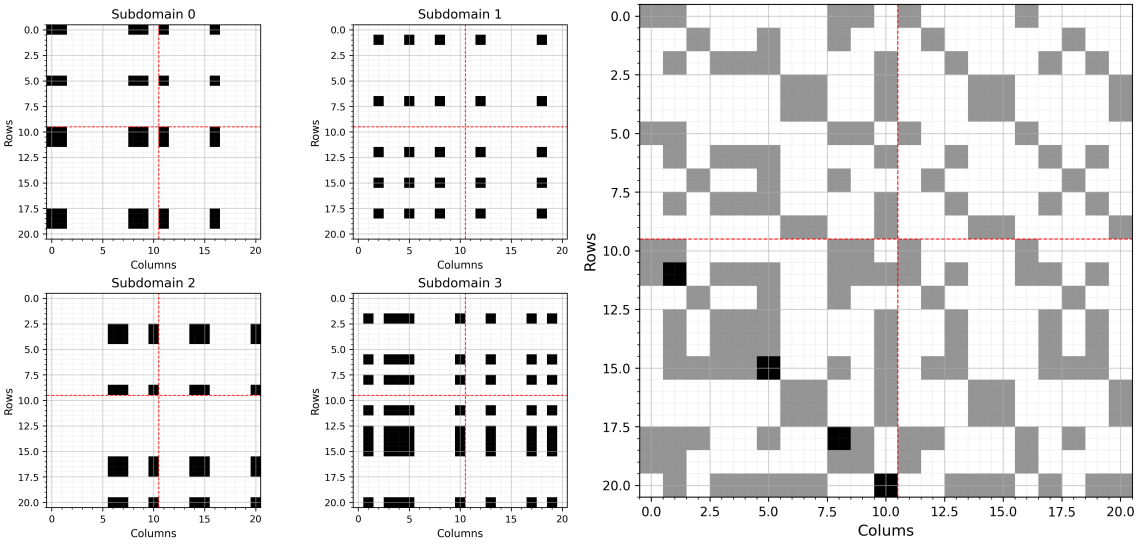
Translating to a Partition of the Jacobian: Gas Networks

The partition of the gas network graph has to be translated to a partition of the corresponding linearized system. Each node in the graph results in either 0 or 1 equation in $\mathbf{F}^g$ and unknown in $\mathbf{x}^g$, depending on the node type. The node types of gas networks are described in Section 3.1.3. A slack node does not have a corresponding nodal equation, since the pressure is specified beforehand, whereas each load node leads to one nodal equation in $\mathbf{F}^g$ and unknown in $\mathbf{x}^g$. All the links of a gas network have unknown gas mass flow, and are thus all of the same type. As a result, each link in the subdomain corresponds to one equation in $\mathbf{F}^g$ and unknown in $\mathbf{x}^g$. The partition of the gas network graph is translated to the linearized system by taking those elements of the global Jacobian which correspond to the equations and unknowns associated with the nodes in that subdomain. Thus, global Jacobian element $\partial \mathbf{F}_i^g / \partial \mathbf{x}_j^g$ is only included in the sub-system Jacobian matrix when the node or link corresponding to equation $\mathbf{F}_i^g$ as well as the node or link corresponding to unknown $\mathbf{x}_j^g$ are part of the subdomain.

The node types and number of adjacent links are not considered when constructing the partition. When the subdomain consists of n nodes, the minimum number of links is $n-1$ in the case of a tree (assuming connectedness). Assuming that the gas network graph is connected, there is at least one link that connects the subdomain to an adjacent subdomain. Therefore, the minimum size of the sub-system Jacobian is $(2n-1)\times(2n-1)$ (assuming that a subdomain contains at most one slack node). The maximum size of the sub-system Jacobian depends on the number of outgoing links to adjacent subdomains. At maximum, all m links of the global network graph are included in a subdomain, even if not all nodes are included. The maximum size of a sub-system Jacobian is therefore $(n+m)\times(n+m)$.

The vector $\mathbf{F}^g$ consist of two parts, first the nodal conservation laws corresponding to the nodes, and next the link flow equations, which correspond to the links. The vector $\mathbf{x}^g$ also consists of two parts, but first contains the unknown link flows, corresponding to the links, and second the unknown nodal pressures, corresponding to the nodes. As a result, the indices of the global Jacobian that make up the sub-system Jacobians are not the same for the rows and columns. In Figure 8.2 the translation of a nodal partition of GasLib-11 into 4 subdomains without overlap to the global Jacobian is visualized. It can be seen in Figure 8.2a that the blocks included in each sub-system Jacobian are not necessarily symmetric, as a result of the node/link order for the rows and columns. The total sub-system Jacobians are always square. In Figure 8.2b, the global Jacobian is visualized, where a grey entry implies that the entry is contained in one subdomain. The darker colour of several entries in the left lower block

(link-link block) indicate that these are contained in 2 or more subdomains. These are exactly the links that connect two different subdomains. Note that this pattern is also not symmetric.



(a) Entries of the global Jacobian included in each subdomain Jacobian.

(b) Entries of the global Jacobian taken into account in a subdomain. Darker entries imply that an entry is included in two or more subdomains.

Figure 8.2: Subdomain inclusion pattern of the global Jacobian, indicating which entries are included in in at least one subdomain as a result of partitioning the GasLib-11 graph into 4 subdomains using the `PETSc` ParMETIS algorithm, without overlap. The red lines indicate a change from nodal to link equations for the rows, and link to nodal parameters for the columns.

Translating to a Partition of the Jacobian: Coupled Networks

The partition of the coupled network graph has to be translated to a partition of the corresponding linearized system. This is done by taking those elements of the global Jacobian which correspond to the equations and unknowns associated with the nodes and links in that subdomain. The structure of the Jacobian of a coupled gas-electricity network is described in Section 5.1.2. Note that the Jacobian matrices of the SC components in a coupled network, J_{gg} and J_{ee} are not necessarily the same as the Jacobian of the SC when solved separately. This is due to a change of the node types as a result of the coupling. The SC Jacobian components J_{gg} and J_{ee} are thus not necessarily square, so the indices of the global Jacobian that make up the sub-system Jacobians are not the same for the rows and columns.

The first subdomain of a coupled network consists of all gas nodes and links, the gas dummy links and the coupling nodes. To construct the sub-system Jacobian associated with the gas subdomain, we thus take the rows of the global Jacobian matrix associated with all gas nodal and link equations and all coupling equations. For the columns, we take exactly those column indices of the global Jacobian matrix associated with all gas nodal and link variables, as well as the gas dummy link variables. With the overlap, the electrical dummy links and electrical nodes associated with that overlap-level are also included. Therefore, the rows of the global Jacobian matrix associated with the included electrical nodal equations are added. For the columns, exactly those columns associated with the included nodal electrical variables are added, as well as the columns associated with the coupling real powers.

The second subdomain of a coupled network consists of all electrical nodes, the electrical dummy links and the coupling nodes. To construct the sub-system Jacobian associated with this electrical subdomain, we take the rows of the global Jacobian association with all electrical nodal equations and all coupling equations. For the columns, we take exactly those indices associated with all electrical nodal variables. With the overlap, the gas dummy links, as well as the gas nodes and links associated with the included gas network components are also included. Therefore, the rows of the global Jacobian matrix corresponding to the gas nodal and link equations are added. For the columns, the indices association with the overlap gas nodes and links are added, as well as the column indices associated with the coupling gas mass flows.

8.2.4. Additive Schwarz Preconditioner Construction

In the codebase, the AS preconditioner is constructed according to (6.14), by computing the action of the preconditioner on an arbitrary vector, as described in Algorithm 6. In this way, the full computation of the preconditioned matrix is not necessary, and computation time is improved by avoiding matrix-matrix multiplications. In the codebase, the AS preconditioner action on an arbitrary vector is passed to the selected `SciPy` solver as an operator.

The local systems corresponding to each subdomain consist of a set of nodes (nodes and links for gas networks), as determined by the chosen partitioning algorithm. During the construction and application of the AS preconditioner, these local systems have to be solved. The local solve of the i-th subdomain consists of solving

$$A_i \mathbf{x}_i = \mathbf{b}_i.$$

This can be done using a direct solver, or again an iterative solution method. In practice, a direct solver such as LU decomposition is often used, especially for smaller subdomains where a direct solve is usually most efficient and accurate. In the codebase, an LU decomposition is used for the local solution step. This factorization is computed and stored beforehand to prevent repeated computation of the factorization and thus improving total runtime.

Three different variants of the AS preconditioner are implemented. First is the basic AS preconditioner, given by (6.14). The second version is the interpolated AS preconditioner, see (6.15), where each entry in the preconditioner is scaled by the number of subdomains it is included in to prevent over-correcting the overlap entries. Finally, the restricted additive Schwarz (RAS) preconditioner can be used, which is implemented as (6.16).

Additive Schwarz Preconditioner Construction in `PETSc`

The AS preconditioner is also a preconditioner option in the `PETSc` library. One can choose between the variants 'Basic', 'Restrict' and 'Interpolate'. The 'Basic' variant refers to the standard AS procedure, as in (6.14). The option 'Restricted' uses the definition of (6.16), and 'Interpolate' refers to the AS variant where weights are included. When using the `PETSc` AS preconditioner, it is possible to only provide the number of subdomains, so that a built-in partitioning method is used to decompose the global Jacobian matrix into subdomains. Alternatively, one can provide a custom partition of the global Jacobian to use. Note that the built-in partitioning of the AS preconditioner in `PETSc` is constructed based on the Jacobian matrix, not based on the energy network graph. The default local solver used is one step of the ILU factorization, but can be set to any built-in direct or iterative solution method.

8.3. Methodology

This section describes and formalises the methodology used to study the AS preconditioned iterative solvers. First, the setup of the various experiments is discussed, focussing on solver configuration. The section concludes with a discussion of the parameters and metrics that are varied in this research to analyse their impact on the AS preconditioner performance.

8.3.1. Experiment Setup

This section includes the setup of the conducted experiments, focussing on the linear and non-linear solver configuration and the initial iterate.

Non-Linear solution method

The non-linear solution method used in this report to determine an $\mathbf{x}_k$ such that $\mathbf{F}(\mathbf{x}_k) \approx 0$ is the NR method, which is described in more detail in Section 5.1.1. During each NR iteration k, the non-linear system is linearised. The Jacobian matrix J of $\mathbf{F}(\mathbf{x}_k)$ is computed, using current iterate $\mathbf{x}_k$. For the stopping criterium the residual $\|\mathbf{F}(\mathbf{x}_k)\|_2$ is used. The method terminates when the residual is smaller than the pre-set tolerance of $1.00 \cdot 10^{-6}$ or when the maximum number of iterations is reached.

Linear solution method

The linear system to be solved during each non-linear NR iteration is given as

$$J(\mathbf{x}_k)\Delta\mathbf{x}_k = -\mathbf{F}(\mathbf{x}_k),$$

as derived in Section 5.1.1. Using the general linear system notation $A\hat{\mathbf{x}} = \mathbf{b}$ for the linear solution step, we thus have $A := J(\mathbf{x}_k)$, $\hat{\mathbf{x}} := \Delta\mathbf{x}_k$ and $\mathbf{b} := -\mathbf{F}(\mathbf{x}_k)$. The linear solver used in this report is restarted GMRES(m). The GMRES method is described in more detail in Section 5.3.2. When a preconditioner is used, it is applied from the left. The restart value is set to $m = 100$, so that a new orthonormal basis of the Krylov subspace is constructed after at most $m = 100$ inner iterations, based on the most recent iterate $\hat{\mathbf{x}}_m$. The maximum number of restart cycles is set to $n = 100$. Therefore, the total maximum number of iterations is $n \times m = 10.000$. The linear solver terminates when one of the following criteria is met:

- The total maximum number of iterations of $n \times m = 10.000$ is reached.
- When the true residual is smaller than the preset absolute tolerance ϵ_a so that $\|\mathbf{r}_m\|_2 = \|\mathbf{b} - A\hat{\mathbf{x}}_m\|_2 \leq \epsilon_a$. In this research, the absolute tolerance is set to $\epsilon_a = 1.00 \cdot 10^{-6}$. The true residual is only computed after each restart cycle, not during, due to high computational costs of evaluating the true residual. The solver can therefore only terminate based in this criterium after a restart occurs.
- When the estimated preconditioned relative residual is smaller than an adaptive relative residual ϵ_r so that $\frac{\|M^{-1}\mathbf{r}_m\|_2}{\|M^{-1}\mathbf{b}\|_2} \leq \epsilon_r$. The adaptive relative tolerance is initially set to $1.00 \cdot 10^{-6}$, but is updated after restart cycle based on whether the inner solve failed to converge after m iterations or when the inner solve converged, but the true residual has not yet reached absolute tolerance ϵ_a. The aim of this adaptive relative tolerance is to reduce the true residual, but to avoid over-solving during each restart cycle. After each restart cycle, the tolerance ϵ_r is multiplied by the minimum of a relative tolerance factor r, initially set to 1, and the relative absolute tolerance. When the restart cycle terminated, but the true residual did not yet sufficiently converge, the relative tolerance factor r is tightened by multiplying with a factor 0.25. When the restart cycle terminated and the true residual converged sufficiently, the relative tolerance factor r is relaxed by multiplying by 1.5, aiming to prevent over-solving during the next restart cycle.

Initial iterate

The initial iterate when solving steady-state load flow for a gas network consists of the initial link gas mass flows and initial nodal pressures. The link gas mass flows are initiated as $q_i = 0.1$, such that each link has a flow of 0.1 per unit. This initial value is chosen so that the link flow is small, but non-zero. The nodal pressures are initiated as an index-based sequence that starts as 0.95 per unit and decreases linearly to 0.90 per unit. The node that got assigned index 1 will thus have an initial pressure of 0.95 per unit, and the node with the highest index an initial pressure of 0.90 per unit. This sequence of initial values is chosen so that the initial nodal pressures are close to the reference pressure but to prevent two equal nodal pressures. When two connected nodes have the same pressure, their pressure difference is exactly zero, implying that the link flow between these nodes is initially zero. The initial iterate $\mathbf{x}_0^g$ is thus given as

$$\mathbf{x}_0^g = \begin{bmatrix} \mathbf{q}_0 \\ \mathbf{p}_0 \end{bmatrix} = \begin{bmatrix} 0.1 \text{ p.u.} \\ \vdots \\ 0.1 \text{ p.u.} \\ 0.95 \text{ p.u.} \\ \vdots \\ 0.90 \text{ p.u.} \end{bmatrix}.$$

The initial iterate when solving steady-state load flow for an electrical network consists of initial nodal voltage angles and magnitudes. In this research, a flat start is used, implying that all nodal voltage angles are initiated as $\delta_i = 0$, and all nodal voltage magnitudes as $|V_i| = 1.0$ per unit. The initial iterate $\mathbf{x}_0^e$ is thus given as

$$\mathbf{x}_0^e = \begin{bmatrix} \boldsymbol{\delta} \\ |\mathbf{V}| \end{bmatrix} = \begin{bmatrix} 0 \text{ p.u.} \\ \vdots \\ 0 \text{ p.u.} \\ 1.0 \text{ p.u.} \\ \vdots \\ 1.0 \text{ p.u.} \end{bmatrix}.$$

In the case of coupled networks, the initial iterate consists of three components, the gas network variables, electrical network variables and the coupling variables. The gas and electrical network variables are initialised the same when the networks are not coupled as described above. The coupling variables are initiated as $q_c = 0$ and $P_c = 0$ for all coupling units.

8.3.2. Performance Metrics

This section introduces the parameters (independent variables) that are varied during the experiments to investigate the performance of the AS preconditioner. They include the parameters associated with the AS preconditioner construction, as well as the variable parameters of the datasets of gas-, electricity and coupled networks.

- **Overlap level**: The level of overlap is a fundamental part of the AS preconditioner, and allows for information propagation among the subdomains. From theory, it is expected that a higher level of overlap results in better conditioning of the preconditioned system that are more expensive to compute, compared to lower overlap levels. In this research, the overlap levels will be varied, depending on the network size.
- **Number of subdomains**: The number of subdomains used to construct the AS preconditioner affects the sub-system size and the number of sub-systems to solve. From theory it is known that a higher number of subdomains results in smaller sub-systems that are computationally cheaper to solve, but can less accurately capture global system behaviour and errors, compared to a smaller number of subdomains. In this research, the number of subdomains will be varied, depending on the test network size. For coupled networks, the number of subdomains is by construction 2.
- **AS variant**: The type of AS preconditioner used determines how the sub-system solutions are used to precondition the global system. It affects how information is propagated through the network and effects the accuracy and computational costs required. In this research, the basic, interpolated and restricted variants are considered.
- **Partitioning algorithm**: The partition algorithm refers to method used to partition the network graph into several subdomains. The partitioning algorithms considered in this research are ParMETIS, CHACO and an average method. The average method does not take the graph structure into account and thus functions as a base to get an indication of the effectiveness of considering the network graph topology. Each partitioning algorithm can result in a different set of subdomains. The effect of the partitioning algorithm will be investigated, as well as the effect of the distribution of certain node types among the subdomains.
- **Network configuration**: The configuration and test network data determines the global Jacobian structure and spectrum. Gas and electricity network load flow problems have different underlying physical properties and the AS performance can therefore be different for both network types. Additionally, there are various network components that are modelled differently. By considering test datasets with different configurations of network components, the effect on the AS preconditioner can be determined.
- **Coupling**: For coupled energy networks new parameters are introduced. The types of coupling units considered in this research are P2G and GFG. Both units are modelled differently. Additionally, a change of node type is necessary for a square system when a coupling unit is introduced. In this research, the type of coupling unit, the node type change and the number of coupling units is varied, depending on the test network size and configuration.

The performance of the AS preconditioner is analysed by considering various metrics (dependent variables), which are analysed when changing the parameters mentioned above.

- **Accuracy**: The accuracy of the AS preconditioned solvers are judged by compute and keeping track of the residual during the linear and non-linear iterative solution processes. Most importantly, this indicates whether the methods converge. Additionally, this gives an indication of how accurate the approximate solution is at what iteration and how many iterations are required to reach a predetermined tolerance. In this research, the linear and non-linear residuals are used to indicate the accuracy of the solver.

- **Speed**: The speed of the solver is monitored by keeping track of the number of iterations required for convergence, for the inner as well as the outer loop. Additionally, it is insightful to measure the CPU time of the phases of the solution process. Both of these metrics indicate how fast the solution is computed, and what potential bottlenecks are. In this research, the number of linear and non-linear iterations is used as the main indicator for method speed.
- **Cost**: The amount of memory and number of matrix-vector products required by the solution method indicates how much storage is required and how costly it is to execute the method. These results can also be used to identify potential bottlenecks and areas of improvement. The cost of the methods in this research is analysed by considering the known general cost of steps included in the methods as well as the matrix size.
- **Robustness**: The conditioning of the underlying coefficient matrix is influential on the performance of the methods. Comparing the robustness of solution methods indicates how robust the method is and whether it is sensitive to the matrix conditioning or initialization. The robustness can be indicated by considering condition number, eigenvalues and singular values, but also by varying the initial iterate. This research considers the spread of the singular values, eigenvalues as well as the condition number of the linearised system matrices.
- **Scalability**: The research area of this study includes large scale energy networks. Therefore, it is important to consider the scalability of the solution method, such that it is able to solve large systems of linearized load-flow equations. The algorithmic scalability is therefore determined in this research by comparing the performance of the solution method in terms of speed, accuracy and/or cost for test problems varying in problem size.

9

Additive Schwarz Preconditioning for Single-Carrier Energy Networks

This chapter presents the results of the load flow analysis when applying the additive Schwarz (AS) preconditioner in the solution process of the steady-state load flow equations of single-carrier (SC) energy networks. The methodology and setup of the performed experiments is discussed in Chapter 8, including the performance metrics in Section 8.3.2. First, the results for small-scale energy networks are presented. Section 9.1 contains the experiments with small-scale gas networks, while Section 9.2 discusses the experiments with small-scale electricity networks. Both sections have the same setup, consisting of sections discussing the general performance of the AS preconditioner, the effect of partition-related parameters and finally the effect of the different AS types considered in this research. For gas networks, an additional section discusses the observed effects of certain network components on the AS preconditioner. After the discussion of the small-scale SC network experiments, Section 9.3 presents the results of AS application to large-scale energy networks. This includes both gas and electricity networks. This section also presents the effects of considering a different target matrix for large-scale datasets. Additionally, Appendix D presents an overview of the solver performance of different solver configurations.

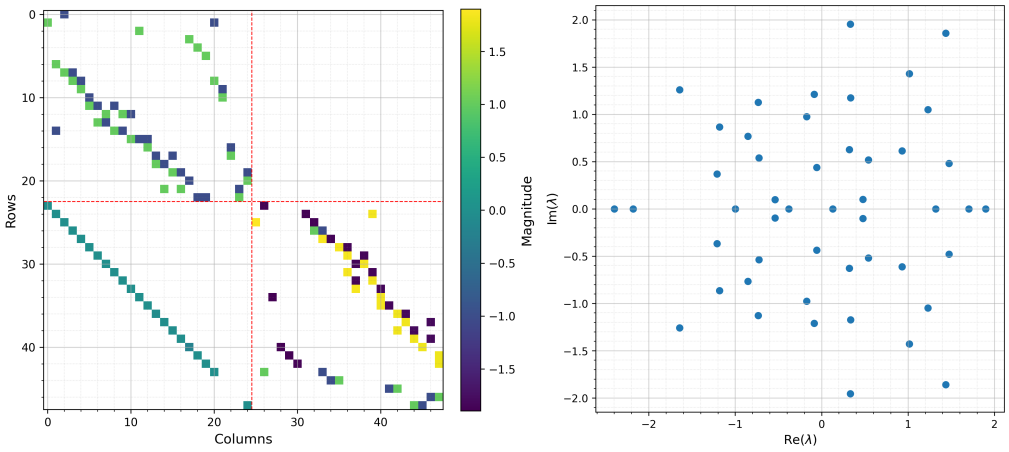
9.1. Small-Scale Gas Networks

This section presents the results of the application of the AS preconditioner in the solution process of steady-state load flow equations of small-scale gas networks. The setup of the experiments and test networks is given in Chapter 8. The gas network test data and configurations consist of several GasLib datasets [53], and are discussed in Section 8.1. The graph presentations of several small-scale networks are given in Appendix B, in Section B.1.

9.1.1. General Effect of AS Preconditioning

The goal of using a preconditioner is to find a system that has the same solution as the original system, but is more efficient to solve than the original system. The effect of the AS preconditioner can thus be analysed by comparing the spectrum and structure of the original Jacobian J with the preconditioned Jacobian $M^{-1}J$. Ideally, the preconditioned Jacobian should resemble a diagonal matrix more closely than the original Jacobian, since a system with a diagonal matrix can be solved efficiently.

The structure of the GasLib-24 Jacobian matrix can be found in Figure 9.1a. It is clear that the Jacobian is of lower block-triangular form, and has the structure as indicated in Equation (5.6). The lower block-triangular structure is not used during the construction of the AS preconditioner. Instead, each sub-system Jacobian is also lower block-triangular. Optionally, one can take advantage of this structure during the solution process of the sub-systems by choice of solution method.



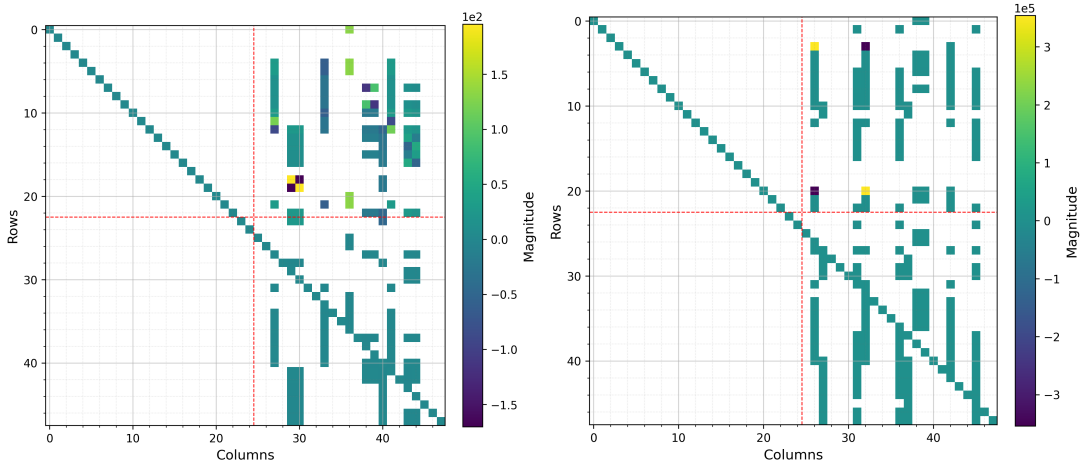
(a) Structure heatmap of the global Jacobian.

(b) Eigenvalue distribution of the global Jacobian.

Figure 9.1: Structure heatmap and eigenvalues of the global Jacobian of GasLib-24, evaluated at initial iterate $\mathbf{x}_0^g$.

Structure of the Jacobian

The structure of the preconditioned Jacobian of the GasLib-24 dataset with the basic AS preconditioner can be found in Figure 9.2. Comparing the figures to Figure 9.1a, one can see that the preconditioned Jacobian matrix shows a more diagonal structure. All elements on the diagonal are non-zero. All other non-zero elements only occur in the right part of the Jacobian, which correspond to the derivatives of $\mathbf{F}^g$ with respect to the nodal pressures. The columns that contain non-zero entries are exactly those associated with the nodal pressures of the nodes adjacent to nodes included in an overlap with another subdomain. The exact location of the non-zero entries in a column is determined by the other subdomain the adjacent overlap node is included in. For example, in Figure 9.2a, columns 27, 33 and 41, that correspond to nodes 3, 9 and 17, have the same non-zero structure, since they are all adjacent to overlap nodes in subdomain 2, see Figure B.2 for the network graph.



(a) 1-level overlap

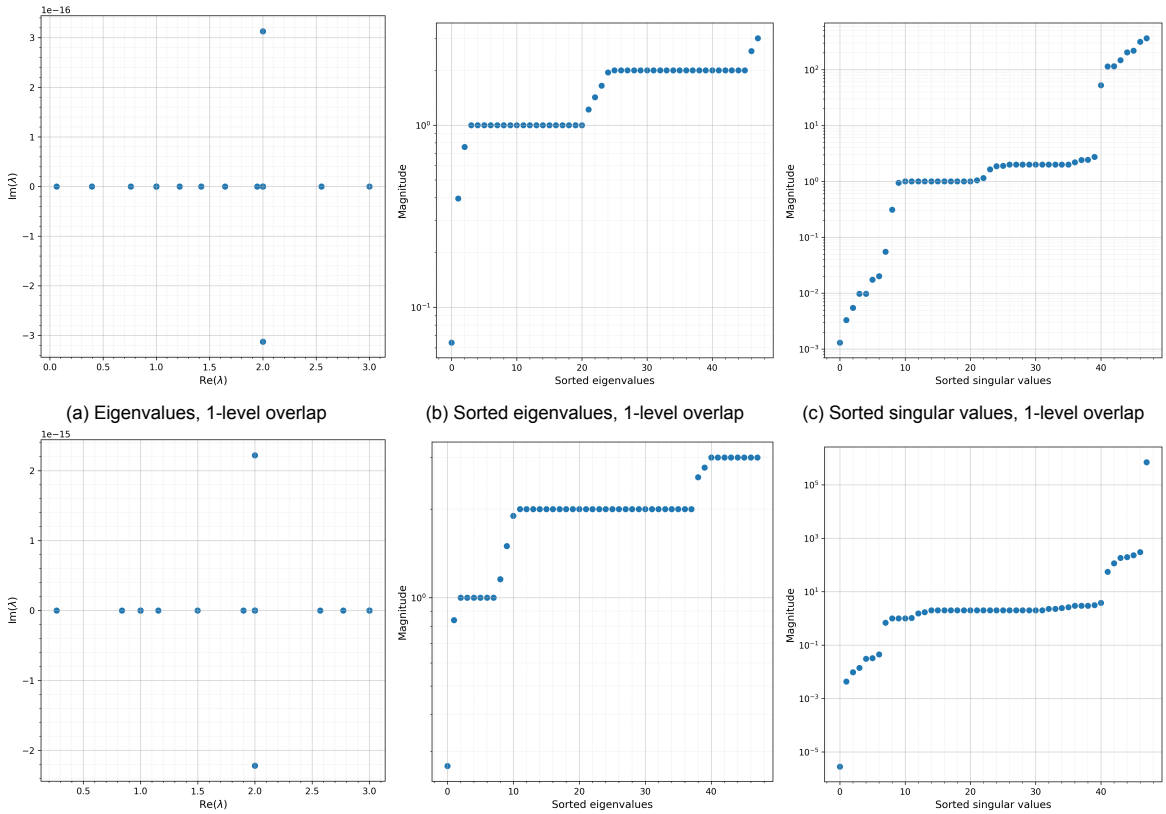
(b) 2-level overlap

Figure 9.2: Structure heatmaps of the global Jacobians of GasLib-24, preconditioned with a basic AS preconditioner for 4 subdomains with ParMETIS, with 1 and 2 level overlaps, evaluated at initial iterate $\mathbf{x}_0^g$.

Spectrum of the Jacobian

The eigenvalues of the original Jacobian matrix of the GasLib-24 dataset can be found in Figure 9.1b. From the plot, one can see that there are eigenvalues with negative real parts, indicating that the original Jacobian matrix is not symmetric positive definite (SPD), and that they are centred around zero. The condition number and the extreme singular values or the original Jacobian as well as various preconditioned Jacobians are given in Table 9.1.

The eigenvalues and singular values of the basic AS preconditioned Jacobians can be found in Figure 9.3. It shows that all eigenvalues of the preconditioned systems have a positive real part. Additionally, the imaginary parts of some of the eigenvalues are very small, and considered negligible. The figures with the sorted eigenvalues show that there are multiple eigenvalues with value 1, 2 or 3. This is the result of certain nodes and links in the gas network graph being included in either 1, 2 or 3 subdomains due to the added overlap. Note that for both overlap cases, there is one smallest eigenvalue that is an order smaller than the second smallest eigenvalue. From the figure, the spectra of the preconditioned Jacobians are distributed more favourably for the iterative linear solution methods employed, compared to the original Jacobian matrix, as they are no longer centred around 0. The figure also contains the singular values for both overlap cases. Most singular values are centred around one. The extreme singular values range from an order of 10^{-2} to 10^{2}. These are likely the result of an imbalance between the updates of the different subdomains. For the 2-level overlap case, the smallest and largest singular value are of more extreme orders, namely 10^{-6} and 10^{5}, respectively. Since the extreme singular values play an important role in the convergence of iterative solution methods, this is not desirable. More on the cause of these extreme singular values can be found in Section 9.1.2.

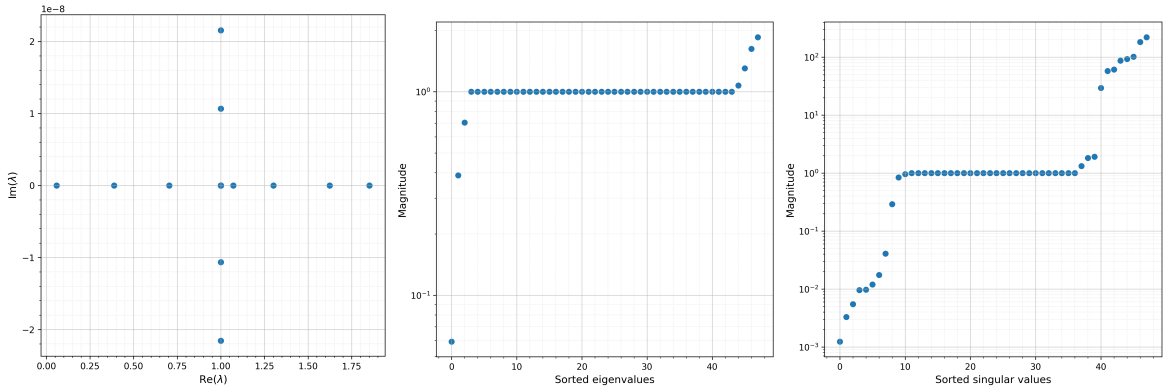


(a) Eigenvalues, 1-level overlap (b) Sorted eigenvalues, 1-level overlap (c) Sorted singular values, 1-level overlap

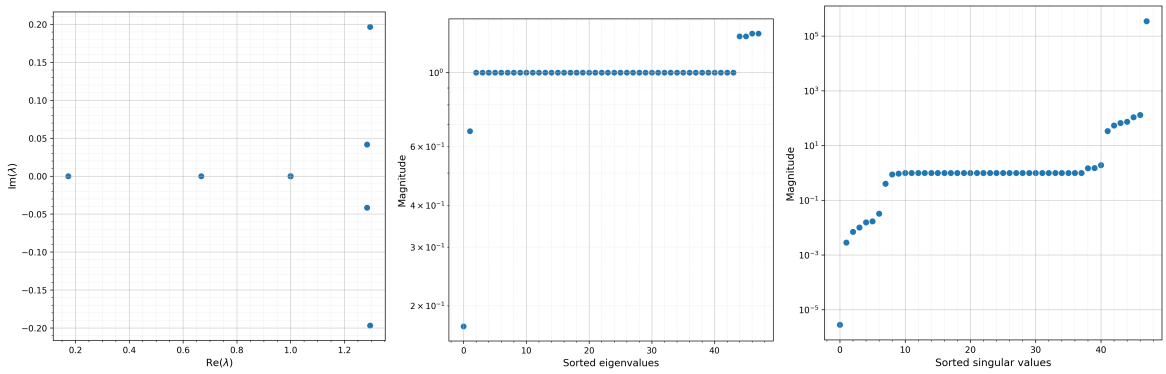
(d) Eigenvalues, 2-level overlap (e) Sorted eigenvalues, 2-level overlap (f) Sorted singular values, 2-level overlap

Figure 9.3: Eigenvalues and singular values of the preconditioned global Jacobian of GasLib-24, preconditioned with a basic AS preconditioner for 4 subdomains with ParMETIS, with 1 and 2 level overlaps, evaluated at initial iterate $\mathbf{x}_0^g$.

The eigenvalues and singular values of the interpolated AS preconditioned Jacobians can be found in Figure 9.4. Similarly to the basic AS case, all eigenvalues have positive real part. As a result of the weighting, the eigenvalues are more clustered around 1, which is different from the basic AS preconditioner in Figure 9.3. The weighting also results in eigenvalues that have a non-zero imaginary part. The singular values show a similar pattern as the basic AS case. The most prominent difference, best visible in the 1-level overlap case, is that in due to the interpolation, the non-extreme singular values are more clustered around 1, whereas for the basic AS the non-extreme singular values also take on the value of 2 or 3. The number of singular values not centred around 1 is the same as when the basic AS preconditioner is applied.



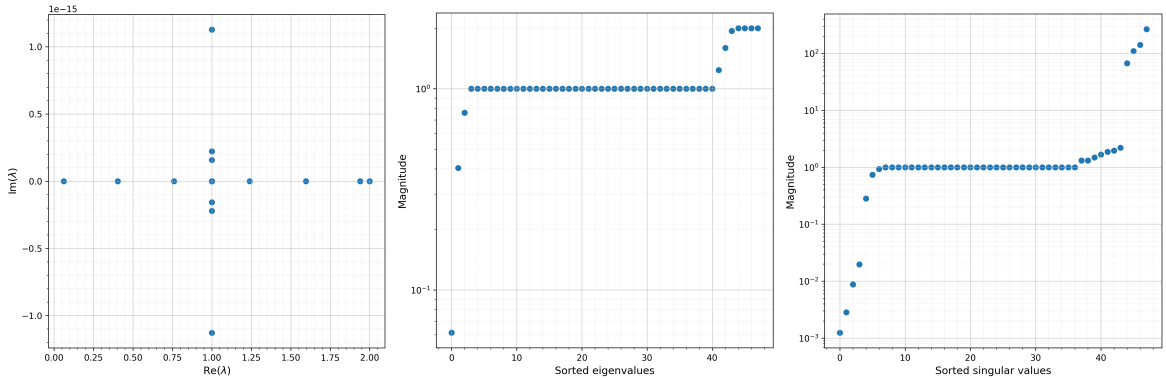
(a) Eigenvalues, 1-level overlap (b) Sorted eigenvalues, 1-level overlap (c) Sorted singular values, 1-level overlap



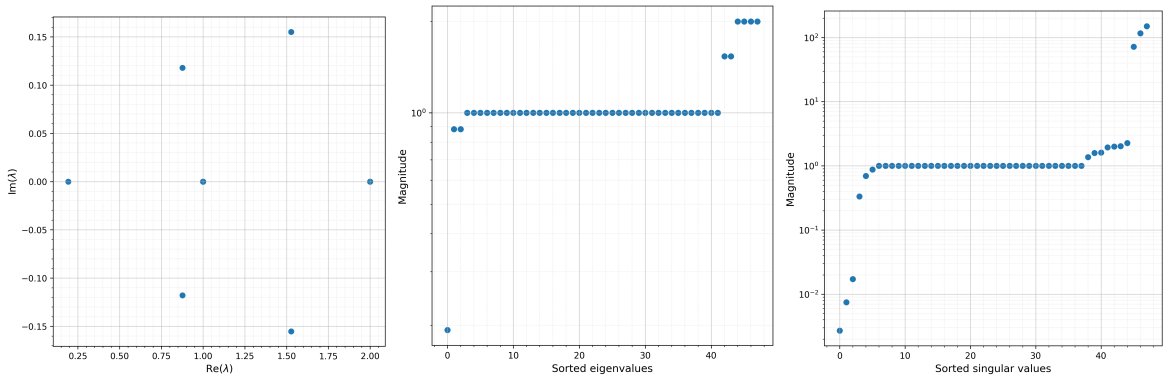
(d) Eigenvalues, 2-level overlap (e) Sorted eigenvalues, 2-level overlap (f) Sorted singular values, 2-level overlap

Figure 9.4: Eigenvalues and singular of the preconditioned global Jacobian of GasLib-24, preconditioned with an interpolated AS preconditioner for 4 subdomains with ParMETIS, with 1 and 2 level overlaps, evaluated at initial iterate $\mathbf{x}_0^g$.

The eigenvalues and singular values of the restricted additive Schwarz (RAS) preconditioned Jacobians can be found in Figure 9.5. Similarly to the basic and interpolated AS cases, all eigenvalues have positive real part. Although the imaginary parts of the complex eigenvalues in the 1-level overlap case are small, they are not negligible. For the 2-level overlap case, the imaginary parts are of the same order as the real parts. The clustering of the eigenvalues and middle singular values is more centred around 1 than when using the basic AS preconditioner, but more spread compared to interpolated AS. In the RAS case, only the internal nodes and links of a subdomain are updated with the local subdomain solve. Therefore, only those elements that are contained in more than one subdomain are updated more than one time. As a result of the used link inclusion method, the doubly updated elements are the gas mass flows through the links that connect one subdomain to another, which explains why there are some eigenvalues and singular values with value 2. In the interpolated AS case, these link updates are also weighted. Compared to the two other AS types, the number of extreme singular values has decreased in the restricted case. An explanation could be that there are less boundary links between subdomains in the prolongation step where a mismatch between subdomain updates can occur. Note that the largest and smallest singular values are no longer of orders 10^5 and 10^{-6}, respectively.



(a) Eigenvalues, 1-level overlap (b) Sorted eigenvalues, 1-level overlap (c) Sorted singular values, 1-level overlap



(d) Eigenvalues, 2-level overlap (e) Sorted eigenvalues, 2-level overlap (f) Sorted singular values, 2-level overlap

Figure 9.5: Eigenvalues and singular values of the preconditioned global Jacobian of GasLib-24, preconditioned with a RAS preconditioner for 4 subdomains with ParMETIS, with 1 and 2 level overlaps, evaluated at initial iterate $\mathbf{x}_0^g$.

The condition number and extreme singular values of the 3 AS types with 1-level and 2-level overlap are given in Table 9.1. Note that in all cases, the AS preconditioner increases the condition number due to the extreme singular values. In Section 9.1.2, more is commented on the high condition numbers of the 2-level overlap cases with the basic and interpolated AS preconditioner.

Table 9.1: The condition number and extreme singular values of the original GasLib-24 Jacobian and the basic, interpolated, and restricted AS preconditioned Jacobians for 4 subdomains with ParMETIS, and varying levels of overlap.

System	AS Type	Condition Number	Largest SV	Smallest SV
Original	-	$4.64 \cdot 10^{2}$	$4.18 \cdot 10^{0}$	$9.00 \cdot 10^{-3}$
1-level overlap	Basic	$2.79 \cdot 10^{5}$	$3.65 \cdot 10^{2}$	$1.31 \cdot 10^{-3}$
	Interpolated	$1.78 \cdot 10^{5}$	$2.21 \cdot 10^{2}$	$1.24 \cdot 10^{-3}$
	Restricted	$2.14 \cdot 10^{5}$	$2.67 \cdot 10^{2}$	$1.25 \cdot 10^{-3}$
2-level overlap	Basic	$2.51 \cdot 10^{11}$	$7.08 \cdot 10^{5}$	$2.82 \cdot 10^{-6}$
	Interpolated	$1.25 \cdot 10^{11}$	$3.54 \cdot 10^{5}$	$2.82 \cdot 10^{-6}$
	Restricted	$6.46 \cdot 10^{4}$	$1.50 \cdot 10^{2}$	$2.71 \cdot 10^{-3}$

Convergence History

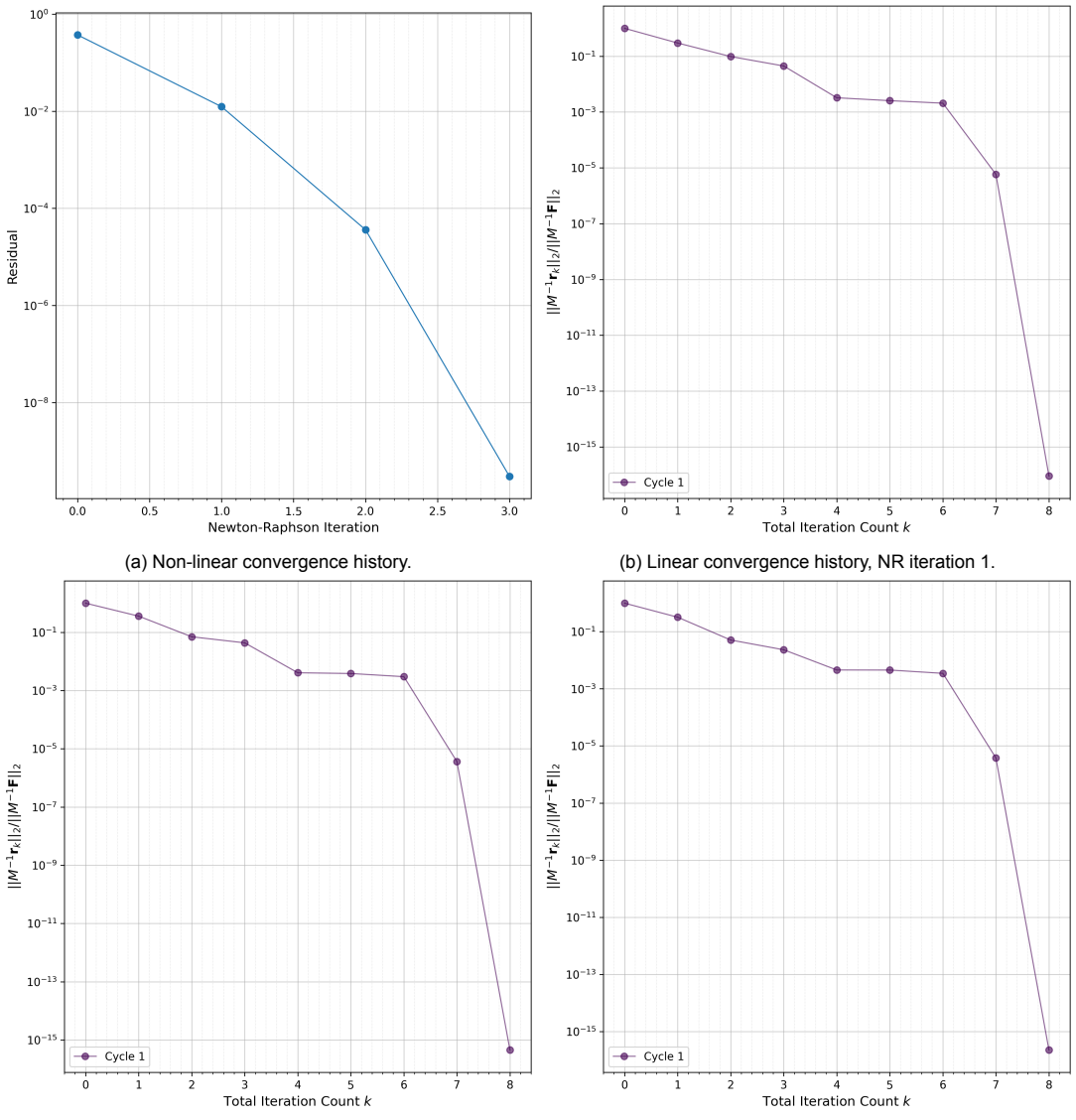
Figure 9.6 contains the linear and non-linear convergence history of the RAS preconditioned GasLib-24 gas network. The RAS preconditioner was constructed using the ParMETIS algorithm with 4 subdomains and a 2-level overlap. For the linear solution steps during each Newton-Raphson (NR) iteration, restarted generalized minimum residual (GMRES) was used. The preconditioned system converged after 3 iterations. The residual $\|\mathbf{F}^g(\mathbf{x}_i^g)\|_2$ at the start of each NR iteration i is also given in Table 9.2.

From Figure 9.6 one can see that in each NR iteration there is super-linear convergence, as is observed often for GMRES. Also note that no restarts were needed during any of the three linear solution

steps.

Table 9.2: Non-linear convergence history of the RAS preconditioned GasLib-24 network, for 4 subdomains and 2-level overlap. Iteration 0 refers to the initial residual.

NR iteration	Residual	GMRES iterations
0	$3.731 \cdot 10^{-1}$	-
1	$1.239 \cdot 10^{-2}$	8
2	$3.612 \cdot 10^{-5}$	8
3	$2.994 \cdot 10^{-10}$	8



(a) Non-linear convergence history.

(b) Linear convergence history, NR iteration 1.

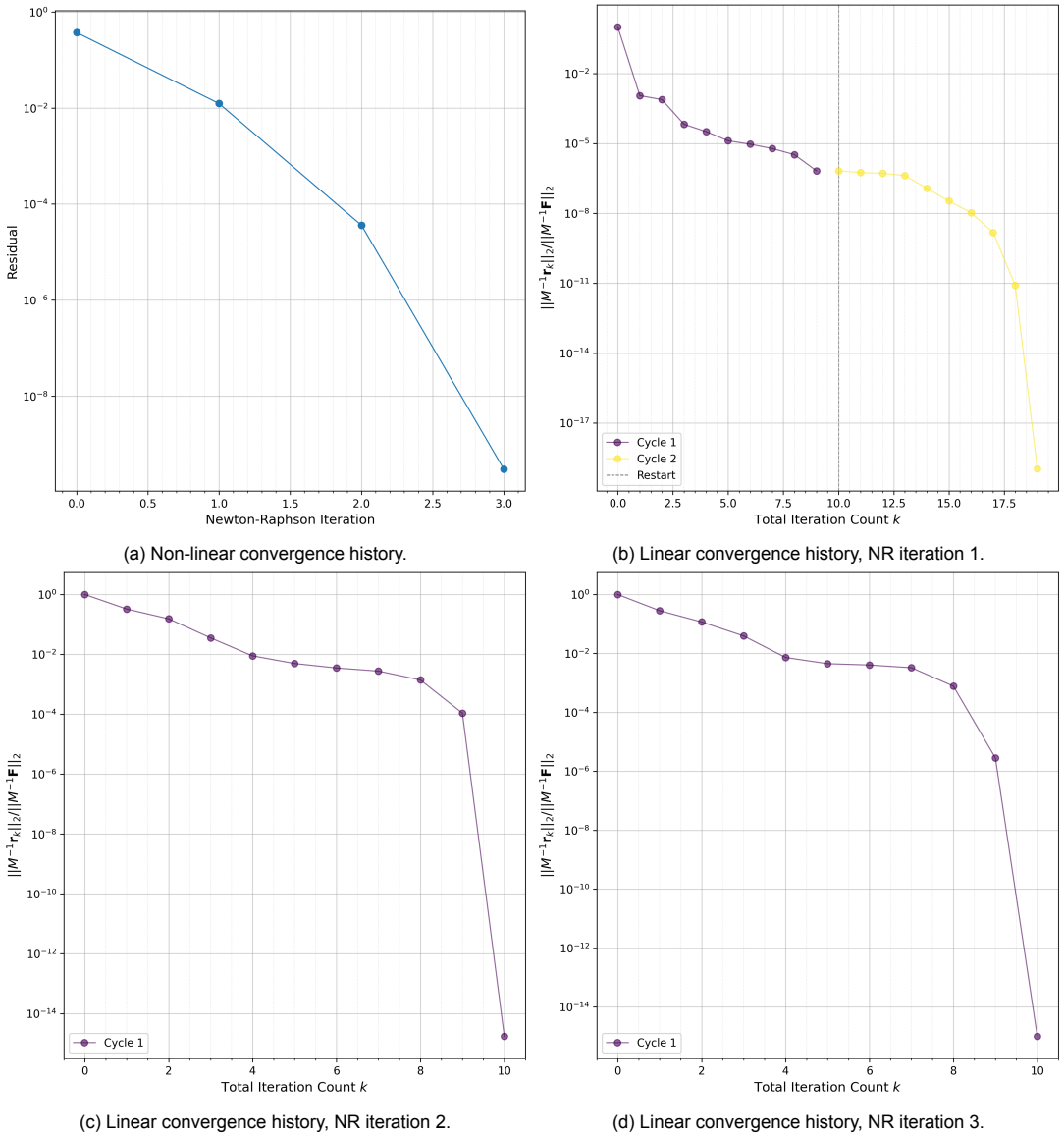
(c) Linear convergence history, NR iteration 2.

(d) Linear convergence history, NR iteration 3.

Figure 9.6: Convergence history of GasLib-24, preconditioned with a RAS preconditioner for 4 subdomains with ParMETIS, with 2 level overlap, using restarted GMRES as linear solution method.

In Figure 9.7, the convergence history of the GasLib-24 gas network is shown, when the basic AS preconditioner is applied, using a partition from the ParMETIS algorithm with 4 subdomains and a 2-level overlap. Comparing the convergence history to the RAS case, the linear convergence history of the first NR iteration differs, since a restart occurred. After the restart, the same super-linear convergence behaviour can be seen. The preconditioned Jacobian matrix of the first NR iteration in this case has

one very large and one very small singular value, see Figure 9.3f. In the first GMRES cycle, they prevent further convergence. After the restart, the noise of these extreme singular values is no longer dominating the search direction, allowing for further convergence.



(a) Non-linear convergence history.

(b) Linear convergence history, NR iteration 1.

(c) Linear convergence history, NR iteration 2.

(d) Linear convergence history, NR iteration 3.

Figure 9.7: Convergence history of GasLib-24, preconditioned with a basic AS preconditioner for 4 subdomains with ParMETIS, with 2 level overlap, using restarted GMRES as linear solution method.

9.1.2. Effect of Network Elements

Gas networks consist of various elements, see Section 2.2. The underlying physical laws per link can thus differ, depending on the element that it represents. In the GasLib datasets, link elements are either a gas pipe, compressor, resistor, short pipe or control valve. In the employed codebase, the links are either modelled as a high-pressure gas pipe, compressor or resistor. The elements representing short pipes and (control) valves are modelled as resistors, with pipe constant $C^g = 1$ (unscaled). It is observed that certain network elements can significantly affect the performance of the AS preconditioner, by worsening its conditioning or preventing its construction.

Resistors

During the construction of the AS preconditioner, various sub-matrices of the Jacobian are computed, that correspond to a selection of nodes and links of the global gas network graph. As a result, there

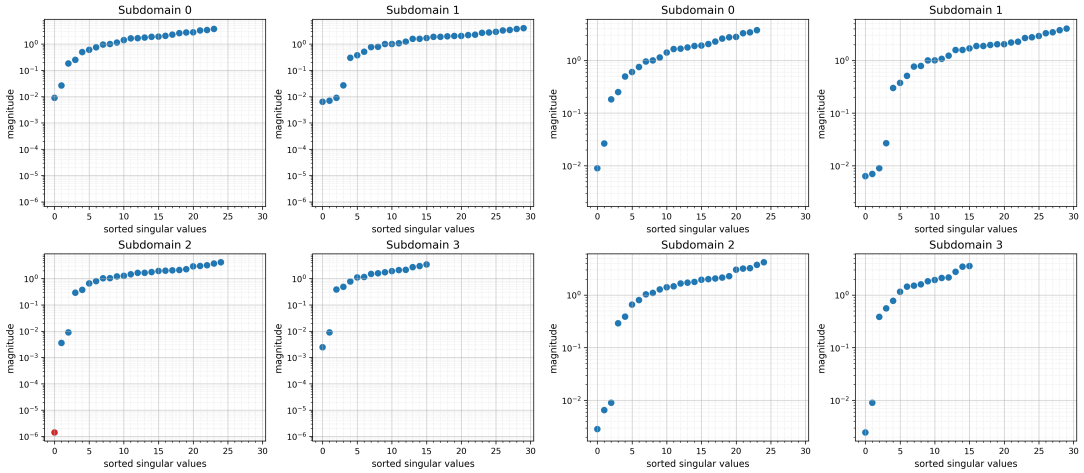
are links included such that one connected node is also part of the subdomain, while the other is not. The pressure at the included node is taken into account during the subdomain solve, while the other nodal pressure remains constant. It can thus occur that the pressure difference along the link, and thus the gas mass flow, can increase or decrease beyond standard values and/or flip defined flow direction. The mismatch between updates of the same network elements from different subdomain solves can influence the conditioning of the AS preconditioned Jacobian.

The basic AS preconditioned Jacobian applied to the GasLib-24 dataset, that consists of 4 subdomains following from the ParMETIS algorithm and a 2-level overlap, shows a very high condition number for the initial guess $\mathbf{x}_0^g$, see Table 9.1 and Figure 9.3f. The order is not comparable to the order of the condition numbers when a 1-level overlap is used, using more or less subdomains or applying a RAS preconditioner.

Figure 9.8a shows the singular values of the sub-system Jacobian matrices of each subdomain when the GasLib-24 network graph is partitioned into 4 subdomains with the ParMETIS algorithm, using a 2-level overlap. It can be seen that subdomain 2 contains a very small singular value, with an order of 10^{-6}, while the smallest singular values following from the other subdomains are of order 10^{-3}. Subdomain 2, with a 2-level overlap, contains node 9, which is connected to 2 boundary links, both modelled as resistors. During the preconditioner construction, an update of subdomain 2 is computed. The pressure of node 9 is updated, while the pressures of the other nodes connected to the two boundary links are not. Since both boundary links are modelled as resistors, they are more sensitive to pressure differences. As a result, the update in link flows through the boundary links has to be of a much larger order than the other links in the subdomain, eventually causing the very large condition number in the sub-system Jacobian.

Figure 9.8b shows the singular values of the sub-system Jacobian matrices of each subdomain when the GasLib-24 network graph is partitioned into 4 subdomains using the ParMETIS algorithm, using a 2-level overlap, but where gas link 3 is modelled as a high-pressure gas pipe instead of a resistor. It is clear that the smallest singular value is no longer of a different order compared to the other subdomains, highlighting that the sensitivity of the two resistors to the pressure differences is the root issue of the high condition number.

Note that when both links are modelled as resistors, but when the RAS preconditioner is used, this issue does not arise, since the link flow through boundary links is not included in the prolongation step. When the interpolated AS preconditioner is used, the effect of the large update is dampened. During the prolongation step, the average of the updates from all subdomains is used. In this case the link is included in 2 subdomains, and the large update is thus roughly halved. This effect can also be seen in the magnitude of the largest and smallest singular values, that are roughly a half of the basic AS preconditioner, see Table 9.1. To avoid errors in the boundary link updates of the individual subdomains from propagating to the AS preconditioner, it is recommended to use the RAS type.



(a) Gas link 3 modelled as a resistor.

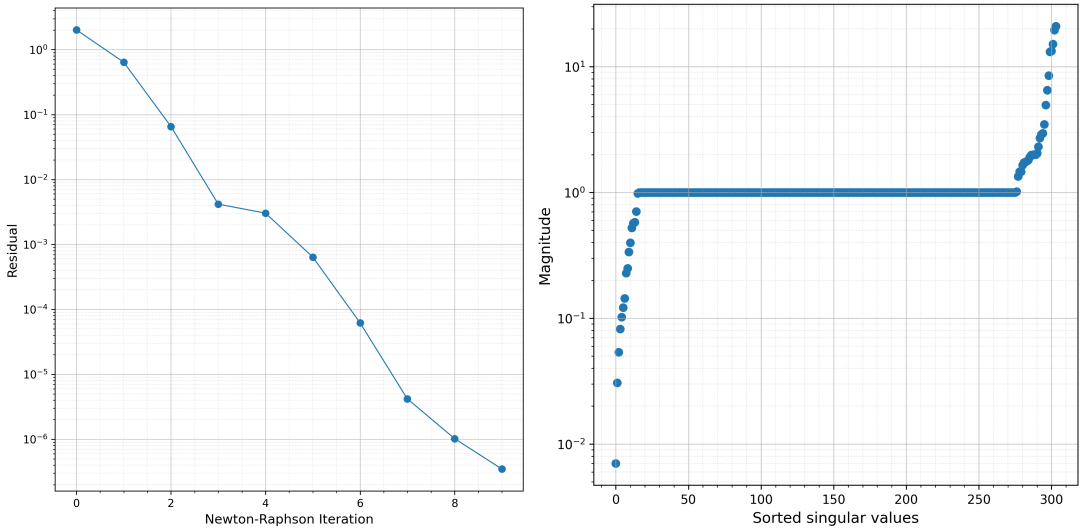
(b) Gas link 3 modelled as a high-pressure gas pipe.

Figure 9.8: Singular values of each subdomain of GasLib-24, partitioned into 4 subdomains with the ParMETIS algorithm and a 2-level overlap. In Figure 9.8a, gas link 3 is modelled as a resistor, while in Figure 9.8b it is modelled as a high-pressure gas pipe.

Compressors

During the application of an AS preconditioner to the first linear solution step for the GasLib-135 dataset, an issue arises. The sub-system Jacobian matrices of subdomains containing certain nodes are singular. As a result, the AS preconditioner cannot be computed and applied. The culprit of the singularity are compressor stations. The link equation used to model a compressor does not contain the gas mass flow along the link. Instead, the nodal pressures of the connected nodes are related through a constant. As a result, $\partial \mathbf{F}_k^L / \partial q_k = 0$ for all compressor links k. When a compressor is a boundary link during the construction of the AS preconditioner, only the nodal pressure of the internal node is included in the subdomain solve. The k-th row of the sub-system Jacobian matrix, $J(\mathbf{x}_i^g)_{k,\cdot}$, corresponding to the link equation of the compressor, has only one non-zero entry, namely $\partial \mathbf{F}_k^L \partial p_j$, where p_j is the nodal pressure of the internal node adjacent to the compressor. When 2 compressors connected to the same node are boundary links as a result of the partition with overlap, the corresponding rows of the sub-system Jacobian matrix are linearly dependent. Therefore, the sub-system Jacobian is singular.

In the GasLib-135 dataset, see Figure B.4 for the network graph, there are 11 nodes with two or more compressor stations connected to it. These 11 nodes lie closely to each other, such that there are only few nodes and links between them. In order for any AS variant preconditioner to be constructed successfully, none of the 11 nodes can be boundary nodes. Unfortunately, in many partitions and overlaps, at least one such node becomes a boundary node. However, it was found that when the ParMETIS algorithm is used to create a partition of 5 subdomains, and a 3-level overlap is applied, all sub-system Jacobian matrices are non-singular. The application of the RAS preconditioner using this partition leads to convergence of the non-linear solver in 9 iterations. The convergence history and singular values of the preconditioned Jacobian matrix are given in Figure 9.9.



(a) Non-linear residual history

(b) Sorted singular values of the preconditioned Jacobian

Figure 9.9: Non-linear residual history and sorted singular values of the preconditioned Jacobian matrix of GasLib-40, partitioned into 5 subdomains with the ParMETIS algorithm, and a 3-level overlap.

9.1.3. Effect of the Partition

An essential part of the application of an AS preconditioner to solving the linearised load flow equations of a gas network is to divide the original problem into multiple smaller overlapping networks. The overlap allows for propagation of information along the sub-networks so that a global solution can be computed. A higher level of overlap thus allows for a more accurate approximation of the global solution and thus better conditioning of the preconditioned Jacobian matrix. However, a high-level overlap results in larger sub-system Jacobian matrices, increasing the cost of computing the preconditioner. The level of overlap should thus be chosen such that there is sufficient propagation of information between the subdomains for effective preconditioning, while minimizing the extra computational costs associated with the overlap. A similar argument holds for the number of subdomains. A high number of subdomains results in multiple smaller sub-systems that are relatively cheap to solve and can be done in parallel. On the other hand, a small subdomain might not be able to sufficiently capture the global system behaviour. This can in turn negatively influence the quality of the preconditioner. When the number of subdomains is small, the computation of each subdomain contribution is computationally expensive, but it more accurately captures the global system behaviour and errors. The number of subdomains should thus be chosen such that each subdomain captures the global behaviour sufficiently well, while minimizing the extra computational costs of each subdomain solve. Additionally, the parallelizability on the employed hardware should be taken into consideration.

Level of Overlap

The effect of the level of overlap is investigated by considering the GasLib-40 gas network data. A partition into 4 subdomains of the network graph is constructed using the ParMETIS algorithm. The RAS preconditioner is then applied, where 1-, 2-, 3- and 4-level overlaps were used. The GasLib-40 dataset is used instead of the GasLib-24 dataset, since the effect of gas network elements on the singular values and convergence behaviour clouds the effect of the level of overlap for the GasLib-24 case, see Section 9.1.2.

The condition number, largest and smallest singular value, number of linear iterations per NR iteration and the relative size of the sub-system Jacobian for each subdomain can be found in Table 9.3. Note that the original Jacobian matrix of the GasLib-40 dataset evaluated at $\mathbf{x}_0^g$ has a condition number of $1.27 \cdot 10^2$. For all levels of overlap included, 5 NR iterations were required. Additionally, the non-linear residual history is similar for each overlap level. The data in the table is in line with what is expected from the theory of the AS preconditioner. The condition numbers of the preconditioned Jacobians decrease when the overlap level increases, indicating a higher quality preconditioner. This

can also be seen in the number of linear iterations required per NR iteration, that generally decrease as a result of a higher overlap level. The bold numbers indicate that a restart occurred during the linear solution process. The table also indicates the range of relative sizes of the sub-system Jacobian matrices. In the case of the 4-level overlap, the largest subdomain consists of 82.1% of the equations and unknowns of the global system. This local subdomain solve comes therefore closer to the cost required to solve the entire global system directly than when a lower level of overlap is used.

Table 9.3: Spectral properties and linear solver performance for the GasLib-40 network, using a RAS preconditioner, constructed with a ParMETIS partition of 4 subdomains and varying levels of overlap. Bold values in the linear iterations column indicate a restart.

Overlap	Cond. Number	Largest SV	Smallest SV	Linear Iterations	Sub-System Size
1-level	$1.16 \cdot 10^4$	$5.75 \cdot 10^1$	$4.95 \cdot 10^{-3}$	11, 11, 11, 11, 10	32.1% - 40.5%
2-level	$6.97 \cdot 10^2$	9.69	$1.39 \cdot 10^{-2}$	9, 10, 10, **16**, 10	40.1% - 53.6%
3-level	$9.88 \cdot 10^1$	7.90	$7.99 \cdot 10^{-2}$	9, 9, 8, 8, 8	54.8% - 69.0%
4-level	$5.77 \cdot 10^1$	7.00	$1.21 \cdot 10^{-1}$	**11**, 8, 8, 8, 8	71.4% - 82.1%

Number of subdomains

The effect of the number of subdomains on the performance of the AS preconditioner is investigated by considering the GasLib-40 network dataset. A partition of the network graph into 2, 4, 6 and 14 subdomains is made using the ParMETIS algorithm. A partition into 14 subdomains was included in this comparison as it is the maximum number of subdomains such that the ParMETIS algorithm does not favour returning empty subdomains over further dividing the nodes. The RAS preconditioner is applied, with a 2-level overlap. The condition number, largest and smallest singular values, number of linear iterations per NR iteration and the relative size of the sub-system Jacobian for each subdomain can be found in Table 9.4. Note that the original Jacobian matrix of the GasLib-40 dataset evaluated at $\mathbf{x}_0^g$ has a condition number of $1.27 \cdot 10^2$. For all numbers of subdomains included, 5 NR iterations were required. Additionally, the non-linear residual history is similar for each number of subdomains. The minimum residual after the first iteration is observed in the case of 4 subdomains, while the maximum residual occurs when 14 subdomains are used.

Table 9.4: Spectral properties and linear solver performance for the GasLib-40 network, using a RAS preconditioner with a 2-level overlap, constructed with a ParMETIS partition of a varying number of subdomains. Bold values in the linear iterations column indicate a restart.

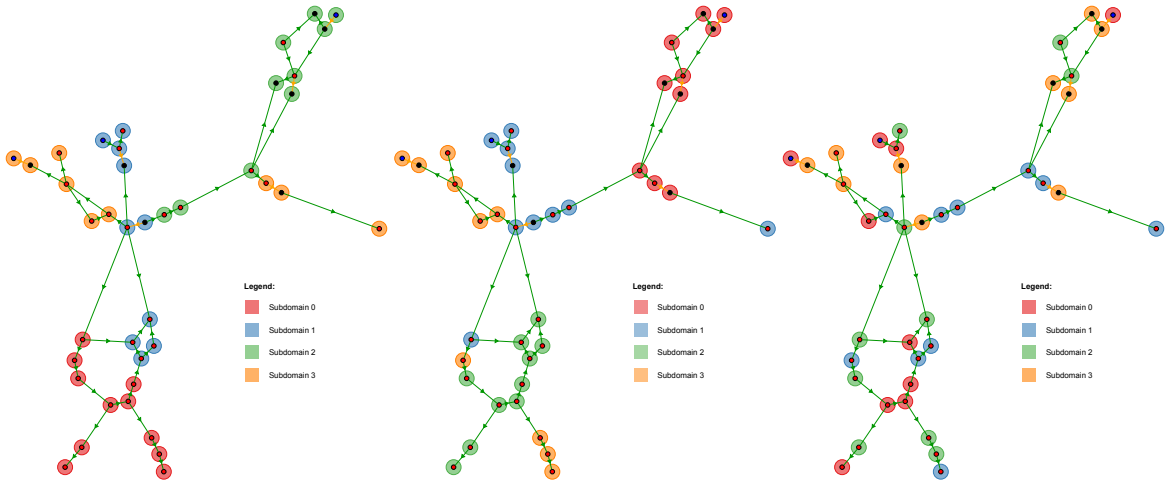
Subdom.	Cond. Number	Largest SV	Smallest SV	Linear Iterations	Sub-System Size
2	$4.97 \cdot 10^2$	$2.12 \cdot 10^1$	$4.26 \cdot 10^{-2}$	4, 4, 4, 4, 4	65.5%
4	$6.97 \cdot 10^2$	9.69	$1.39 \cdot 10^{-2}$	9, 10, 10, **16**, 10	40.1% - 53.6%
6	$1.58 \cdot 10^3$	$2.06 \cdot 10^1$	$1.31 \cdot 10^{-2}$	11, **16**, **21**, 11, 11	27.4% - 57.1%
14	$1.89 \cdot 10^3$	$2.25 \cdot 10^1$	$1.19 \cdot 10^{-1}$	**25**, **33**, **36**, 17, 17	11.9% - 47.6%

The data in the table is in line with what is expected from the theory of the AS preconditioner. The condition number is smallest in the minimum case of 2 subdomains. For this preconditioner, the number of linear iterations required per non-linear iteration is also lowest compared to the other cases. These metrics both indicate a higher quality preconditioner. On the other hand, both subdomains with a 2-level overlap result in sub-system Jacobians consisting of 65.5% of the total number of unknowns, which can make computation of the preconditioner expensive. When the number of subdomains is increased, the condition number of the preconditioned Jacobian also increases. Note that the difference between the largest and smallest condition numbers is of order 10^1. The effect of increasing the number of subdomain is more prominent for the number of linear iterations required during each NR iteration. The bold values indicate that one or more restarts were needed. Not only is there an increase in the number of iterations, but also in the number of restarts. The range of relative sub-system Jacobian size also increases when the number of subdomains increases. If a subdomain contains a relatively little number of nodes, the nodes added to the subdomain by overlap can become dominating in the sub-system Jacobian size. When a node has many adjacent nodes, the overlap can significantly increase the number of nodes within the subdomain. If a large number of subdomains is used, there is more

variation in the topology and connectedness of each sub-system network, resulting in a wider range of relative sub-system Jacobian sizes.

Partitioning Method

The effect of the partitioning method is investigated by considering the GasLib-40 test network data. This dataset is considered since it is small enough to visualise the partition, but large enough that variations in the partitions can occur. When no preconditioner is applied, the GasLib-40 initial Jacobian has a condition number of $1.27 \cdot 10^2$, and requires 84 linear iterations for each of the 5 NR iterations to converge. This is the maximum number of linear iterations required, as it corresponds to the system size. The ParMETIS, CHACO and average algorithms are used to partition the GasLib-40 network graph, see Figure 9.10. The average algorithm does not take the network topology into account. It computes the average number of nodes required per subdomain to distribute the total number of nodes evenly among the subdomains. Then the nodes are added to the subdomain based on their index, such that the first n nodes are included in the first subdomain, and so on. The number of the nodes is such that the slack node is the first node, then come all sources, and finally all sinks.



(a) ParMETIS (b) CHACO (c) Average

Figure 9.10: Network graph of the GasLib-40 dataset, partitioned into 4 subdomains using the ParMETIS, CHACO and average partitioning algorithms.

From Figure 9.10, one can see that the ParMETIS and CHACO partitioning algorithms both contain subdomains with disconnected components. For example, subdomain 3 as a result of the CHACO partitioning algorithm, that consists of three disconnected components. Such a subdomain is more likely to have a higher number of adjacent nodes and links. When an overlap is added during the AS construction, such disconnected components can become connected when overlap is added.

The RAS preconditioner is constructed using the three partitioning methods mentioned above, using 1- and 2-level overlaps. Table 9.5 contains the condition number, number of linear iterations and relative sub-system sizes of the considered cases. For all three partitioning algorithms, the application of all the AS preconditioner variants increased the condition number compared to the original un-preconditioned Jacobian, but improved the number of linear iterations required per NR iteration. Based on the data available in Table 9.5, the ParMETIS and CHACO partitions result in less linear iterations required. After adding overlap, the average method leads to larger relative sub-system sizes, for both the 1- and 2-level overlap cases. This is due to the higher number of total adjacent nodes and links of the subdomains when there are disconnected components. Despite the larger sub-system size, the pre-conditioner performance is not improved, compared to the CHACO and ParMETIS cases. Based on the presented data, there are no large difference between the CHACO and ParMETIS algorithms in terms of preconditioner performance. The condition numbers associated with the CHACO algorithm are lower for both levels of overlap, and, on average, 1 to 2 linear iterations less are required. The ranges of relative sub-system sizes for both algorithms are similar. Particularly for smaller network graphs and

subdomains, the topological connectedness of the individual nodes can dictate the sub-system size rather than the partition itself.

Table 9.5: Spectral properties and linear solver performance for the GasLib-40 network, using a RAS preconditioner constructed with a partition into 5 subdomains using the ParMETIS, CHACO and average partitioning algorithms, with a 1- and 2-level overlap. Bold values in the linear iterations column indicate a restart.

Algorithm	Overlap	Condition number	Linear iterations	Sub-system size
ParMETIS	1-Level	$1.16 \cdot 10^4$	11, 11, 11, 11, 10	32.1% - 40.5%
	2-Level	$6.97 \cdot 10^2$	9, 10, 10, **16**, 10	40.5% - 53.6%
CHACO	1-Level	$2.46 \cdot 10^3$	9, 9, 9, 9, 9	31.0% - 46.4%
	2-Level	$2.54 \cdot 10^2$	10, 10, 9, **10, 11**	33.3% - 56.0%
Average	1-Level	$2.57 \cdot 10^3$	**22**, 15, **32, 24**, 15	60.7% - 71.4%
	2-Level	$3.33 \cdot 10^2$	11, 11, **16**, 10, 10	70.2% - 88.1%

9.1.4. AS Types: Basic, Interpolated and Restricted

The three types of AS preconditioner considered in this research are basic, interpolated and restricted, see Section 6.2.3. They differ in the prolongation step, where the local updates are transferred back to the global space. The basic AS preconditioner is constructed as

$$M_{\text{AS}}^{-1} = \sum_{i=1}^{N} {R_i^{\delta}}^T A_i^{-1} R_i^{\delta}.$$

A local system corresponding to subdomain i, A_i, is computed and solved (directly), after which all components associated with subdomain i are updated. The interpolated AS preconditioner is constructed as

$$M_{\text{IAS}}^{-1} = \sum_{i=1}^{N} {R_i^{\delta}}^T \mathbf{w}_i A_i^{-1} R_i^{\delta}.$$

The weighting vector $\mathbf{w}_i$ contains the global weight of all elements in subdomain i. The weight is determined by the number of subdomains the element is contained in, after adding overlap. This way, the elements that are included in multiple subdomains receive an average update. Finally, the RAS is constructed as

$$M_{\text{AS}}^{-1} = \sum_{i=1}^{N} {R_i^{0}}^T A_i^{-1} R_i^{\delta}.$$

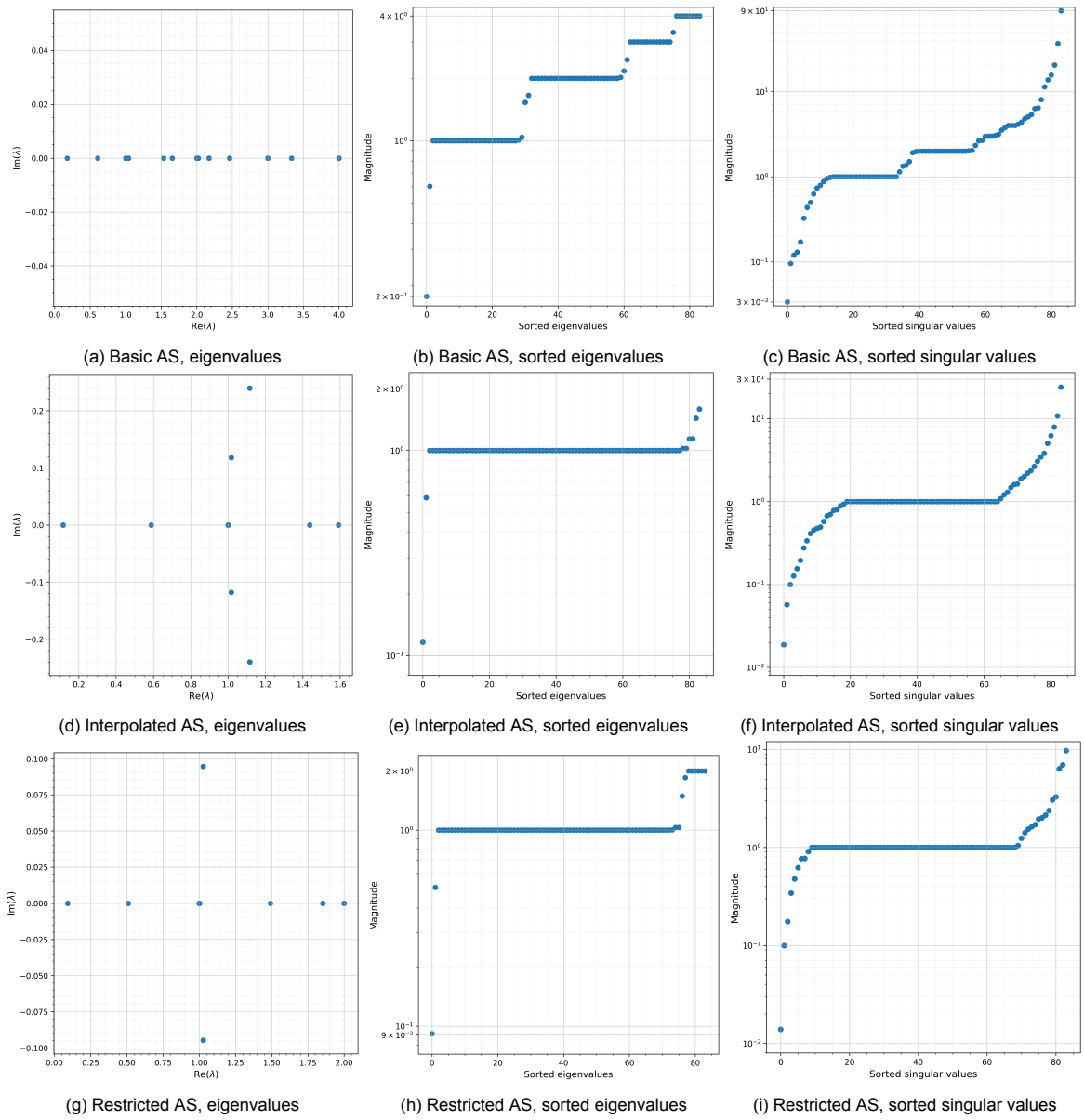
Here, only the elements that are contained in the subdomain without overlap are updated. The variables associated with the overlap are taken into account when computing the update, but are not updated in the global space. The RAS has the advantage that the prolongation step can be done in parallel, as there is no communication necessary between the subdomains. Note that due to the employed link-inclusion method (see Section 8.2, there are gas link elements included in two subdomains, even when there is no added overlap. With this link-inclusion method, there has to be some communication between the subdomains, even if the RAS preconditioner is used. Alternatively, each link that is shared between subdomains can be assigned to one subdomain for the prolongation step.

This section investigates the performance of the three different variants of the AS preconditioner. The effect on the preconditioned Jacobian spectrum, convergence history and error propagation are discussed.

Spectrum of the preconditioned Jacobian

Figure 9.11 gives an overview of the eigenvalues and singular values of the preconditioned Jacobian of the GasLib-40 network, comparing the basic, interpolated and RAS types. The condition numbers, extreme singular values and linear iteration counts are summarised in Table 9.6. For all types, the eigenvalues have positive real part. For the basic AS, all eigenvalues are real, whereas the interpolated and RAS result in some complex eigenvalues. Additionally, the sorted eigenvalue magnitudes of the

basic AS case shows clusters of eigenvalues at 1, 2, 3 and 4. This pattern is also present for the singular values, where multiple singular values are exactly 1, 2, 3 or 4, although there are less of these integer-valued singular values in total compared to the eigenvalues. When the interpolated AS preconditioner is used, all integer-valued eigenvalues and singular values are 1. Finally, the RAS preconditioner also shows most integer-valued eigenvalues and singular values being 1, and a few are 2. This can be explained by noticing that there are several links between the subdomains, that are thus included in 2 subdomains (without overlap). The extreme eigenvalues of the basic AS preconditioned system are slightly more extreme than those of the interpolated and RAS cases.



(a) Basic AS, eigenvalues (b) Basic AS, sorted eigenvalues (c) Basic AS, sorted singular values

(d) Interpolated AS, eigenvalues (e) Interpolated AS, sorted eigenvalues (f) Interpolated AS, sorted singular values

(g) Restricted AS, eigenvalues (h) Restricted AS, sorted eigenvalues (i) Restricted AS, sorted singular values

Figure 9.11: Eigenvalues and singular values of the preconditioned global Jacobians of GasLib-40, with a basic, interpolated and RAS preconditioner for 4 subdomains with ParMETIS and 2-level overlap, evaluated at initial iterate $\mathbf{x}_0^g$.

Table 9.6: Spectral properties and linear solver performance for the GasLib-40 network, using a basic, interpolated and RAS preconditioner, constructed with a ParMETIS partition of 4 subdomains and a 2-level overlap. Bold values in the linear iterations column indicate a restart.

AS type	Cond. number	Largest SV	Smallest SV	Linear iterations	Final residual
Basic	$2.63 \cdot 10^{3}$	$8.89 \cdot 10^{1}$	$3.38 \cdot 10^{-2}$	12, 14, **18**, **24**, **24**	$7.492203 \cdot 10^{-8}$
Interpolated	$1.28 \cdot 10^{3}$	$2.41 \cdot 10^{1}$	$1.88 \cdot 10^{-2}$	11, **18**, 11, **12, 13**	$7.492185 \cdot 10^{-8}$
Restricted	$6.97 \cdot 10^{2}$	$9.69 \cdot 10^{0}$	$1.39 \cdot 10^{-2}$	9, 10, 10, **16**, 10	$7.492076 \cdot 10^{-8}$

Convergence history
Table 9.6 contains the number of linear iterations per NR iteration, as well as the final residual when solving the steady-state load flow equations of the GasLib-40 dataset, using basic, interpolated and RAS preconditioners. The RAS preconditioner has the smallest condition number, indicating a better performing preconditioner. This is confirmed by the number of linear iterations, that is in general lowest for the restricted case out of the three.

Error propagation
When a basic AS preconditioner is constructed, network elements that are included in more than one subdomain (after adding the overlap) are influenced by multiple subdomains at the same time. This can lead to inconsistent corrections of the nodal pressures and link flows, which slows down convergence. When weights are added in the interpolated AS preconditioner, these corrections from the different subdomains are blended more smoothly. This dampens sharp jumps at the boundary nodes and links and can make the preconditioner more robust. In the case of RAS, the update is influenced by only one subdomain. This reduces the exchange of information and errors between subdomains, usually leading to more efficient error damping. These expected performances of the different AS types is reflected in the data of Table 9.6, where the RAS preconditioner improved conditioning most, and required the least number of linear iterations per NR iteration.

Section 9.1.2 discusses the effect of resistors as boundary elements on the AS preconditioner. Not only can the RAS type handle error damping more efficiently, large errors due to sensitive network elements on the subdomain boundary are less strongly propagated to the preconditioner itself. Therefore, the RAS preconditioner is more robust against large boundary errors as a result of partitioning the network graph.

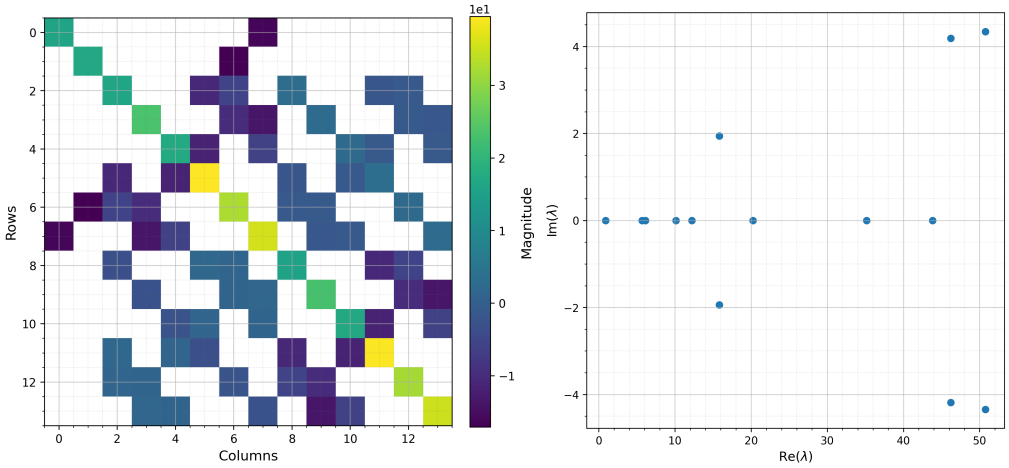
9.2. Small-Scale Electricity Networks

This section presents the results of the application of the AS preconditioner in the solution process of the steady-state load flow equations of small-scale electricity networks. The setup of the experiments and test networks is given in Chapter 8. The electricity network test data and configurations consist of several PandaPower datasets [57], and are discussed in Section 8.1. The graph presentations of several small-scale networks are given in Appendix B, in Section B.2.

9.2.1. General Effect of AS Preconditioning

The aim of applying the AS preconditioner is to find a system that has the same solution as the original system, but is more efficient to solve. The general effect of the AS preconditioner can thus be analysed by considering the structure and spectrum of the preconditioned Jacobian, $M^{-1}J$, and the convergence history when the preconditioner is applied.

The case9 PandaPower network graph representation is given in Figure B.6. The structure of the corresponding case9 Jacobian matrix can be found in Figure 9.12a. Note that the non-zero structure of the Jacobian matrix is symmetric, but that the matrix itself is not symmetric. This also follows from the eigenvalues, given in Figure 9.12b, that all have positive real part and are complex.



(a) Structure heatmap of the global Jacobian.

(b) Eigenvalue distribution of the global Jacobian.

Figure 9.12: Structure heatmap and eigenvalues of the global Jacobian of case9, evaluated at initial iterate $\mathbf{x}_0^e$.

A basic AS preconditioner was constructed by partitioning the case9 network graph into 2 subdomains using the ParMETIS algorithm, see Figure 9.13, and applying a 1-level overlap. The effect of this preconditioner on the Jacobian structure, singular values and eigenvalues can be seen in Figure 9.14.

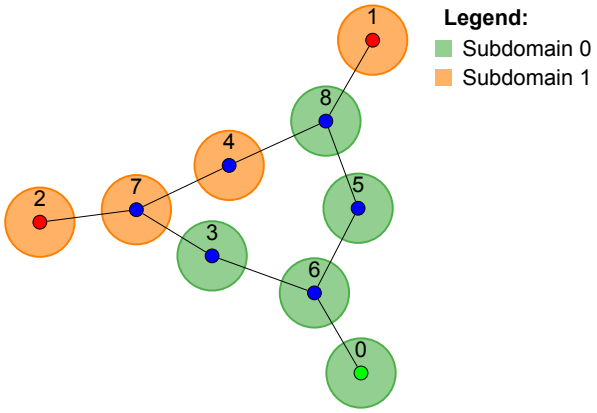
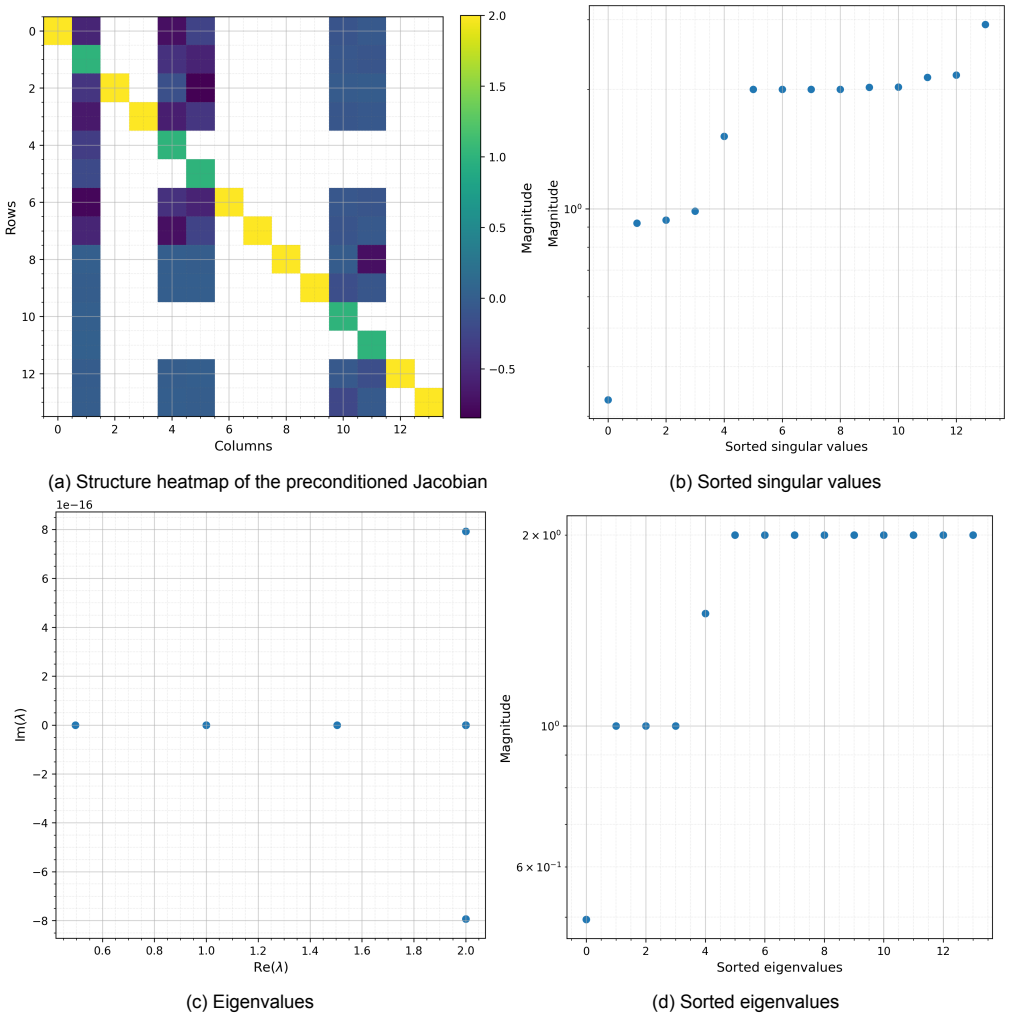


Figure 9.13: Network graph of the case9 dataset. A red node represents a PV node, a blue node represents a PQ node and the slack node is indicated with a green colour.



(a) Structure heatmap of the preconditioned Jacobian

(b) Sorted singular values

(c) Eigenvalues

(d) Sorted eigenvalues

Figure 9.14: Structure heatmap, singular values and eigenvalues of the global Jacobian of case9, preconditioned with a basic AS preconditioner for 2 subdomains with ParMETIS, with 1-level overlap, evaluated at initial iterate $\mathbf{x}_0^e$.

Structure of the Jacobian

The application of the basic AS preconditioner results in a Jacobian with a distinct diagonal structure, see Figure 9.14a. The entries on the diagonal are either 1 or 2. The columns where the diagonal entry is 1 contain non-zero entries. These columns correspond to the variables associated with the nodes adjacent to a node in another subdomain, after including the overlap. The non-zero rows in Figure 9.14a correspond to variables of nodes 2, 5 and 6. Looking at Figure 9.13, node 2 is not included in the overlap of subdomain 0, but is adjacent to node 7, that is part of the overlap. Similarly, nodes 5 and 6 are not included in the overlap over subdomain 1, but are adjacent to nodes 3 and 8 that are included in the overlap.

Spectrum of the Jacobian

The distinction between nodes that are included in 1 or 2 subdomains (with the overlap) can also be seen in the eigenvalues. Figure 9.14d shows the magnitude of the eigenvalues, where it is shown that most eigenvalues have either magnitude 1 or 2. The number of eigenvalues of magnitude 1 or 2 does not appear to be directly related to the number of nodes included in either 1 or 2 subdomains. In Figure 9.14c, one can see that the imaginary parts of the eigenvalues are very small, and considered negligible. Figure 9.14b contains the sorted singular values. Similarly to the sorted eigenvalues, the smallest entry is an outlier, that is an order smaller than all other eigen- and singular values. This indicates that the direction associated with this singular value is the least influential part of the system. The condition number and extreme singular values of the 3 AS types with 1-level overlap applied to the

case9 dataset are given in Table 9.7. In all three cases, the application of the preconditioner results in a smaller condition number.

Table 9.7: The condition number and extreme singular values of the original case9 Jacobian and the basic, interpolated, and restricted AS preconditioned Jacobians for 2 subdomains with ParMETIS.

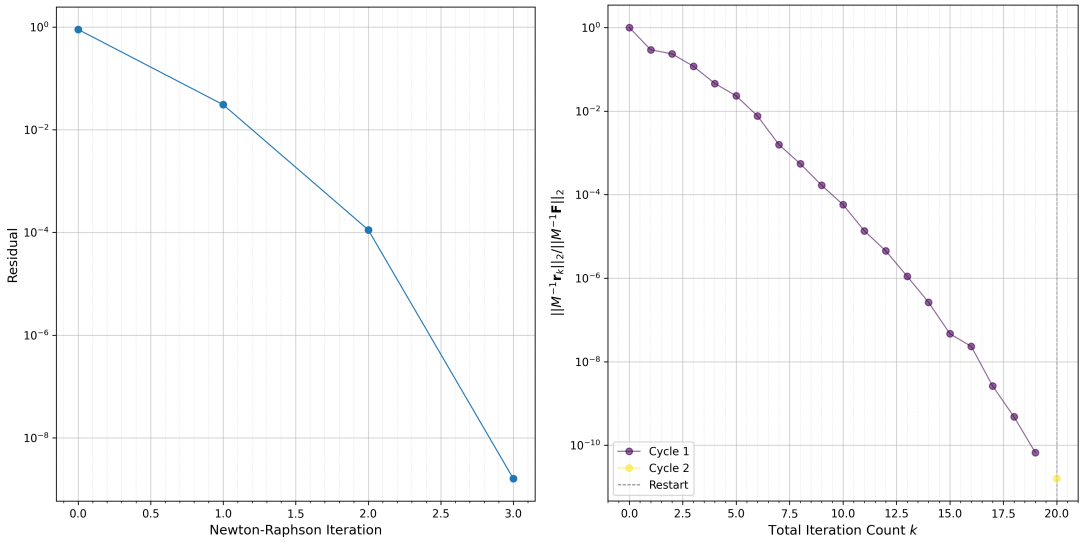
System	AS type	Condition number	Largest SV	Smallest SV
Original	-	$5.94 \cdot 10^{1}$	$5.35 \cdot 10^{1}$	$9.01 \cdot 10^{-1}$
1-level overlap	Basic	$5.85 \cdot 10^{0}$	$2.92 \cdot 10^{0}$	$3.30 \cdot 10^{-1}$
	Interpolated	$5.01 \cdot 10^{0}$	$1.62 \cdot 10^{0}$	$3.23 \cdot 10^{-1}$
	Restricted	$6.10 \cdot 10^{0}$	$2.02 \cdot 10^{0}$	$3.31 \cdot 10^{-1}$

Convergence History

Figure 9.15 contains the linear and non-linear convergence history of the basic AS preconditioned case30 electrical network. The solver configurations and stopping criteria used are described in Section 8.3.1. The preconditioner was constructed using the ParMETIS algorithm with 4 subdomains and a 2-level overlap. For the linear solution steps during each NR iteration, restarted GMRES was used. The preconditioned system converged after 3 NR iterations. The residual $\|\mathbf{F}^g(\mathbf{x}_i^g)\|_2$ at the start of each NR iteration i is also given in Table 9.8. From Figure 9.15, one can see that the non-linear convergence history shows super-linear convergence. During each NR iteration the convergence is also super-linear. One restart occurred at the end of the first NR iteration.

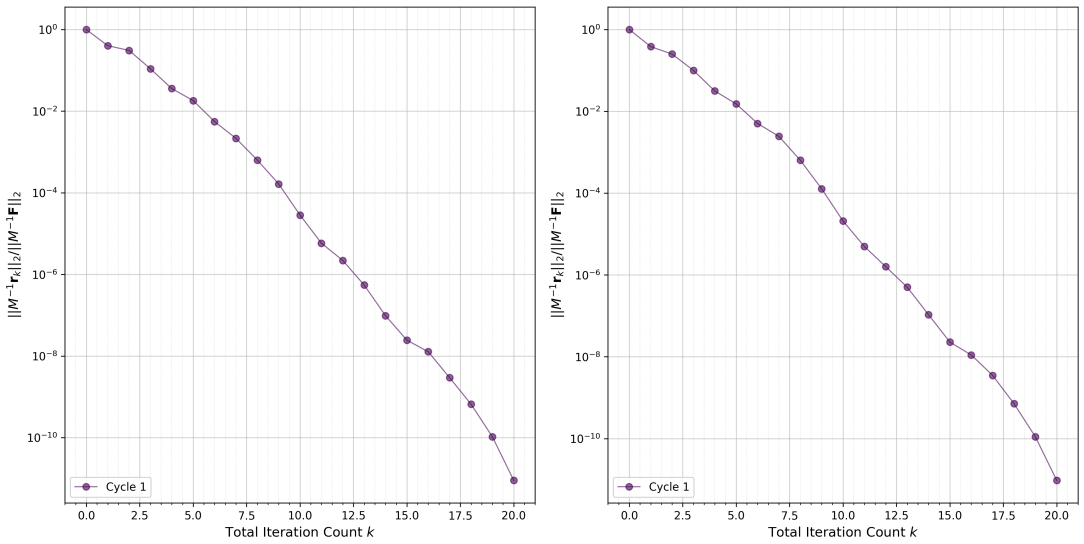
Table 9.8: The non-linear convergence history of the basic AS preconditioned case30 network, for 4 subdomains and 2-level overlap. Iteration 0 refers to the initial residual.

NR iteration	Residual	GMRES iterations
0	$8.980 \cdot 10^{-1}$	-
1	$3.073 \cdot 10^{-2}$	20
2	$1.123 \cdot 10^{-4}$	20
3	$1.603 \cdot 10^{-9}$	20



(a) Non-linear convergence history.

(b) Linear convergence history, NR iteration 1.



(c) Linear convergence history, NR iteration 2.

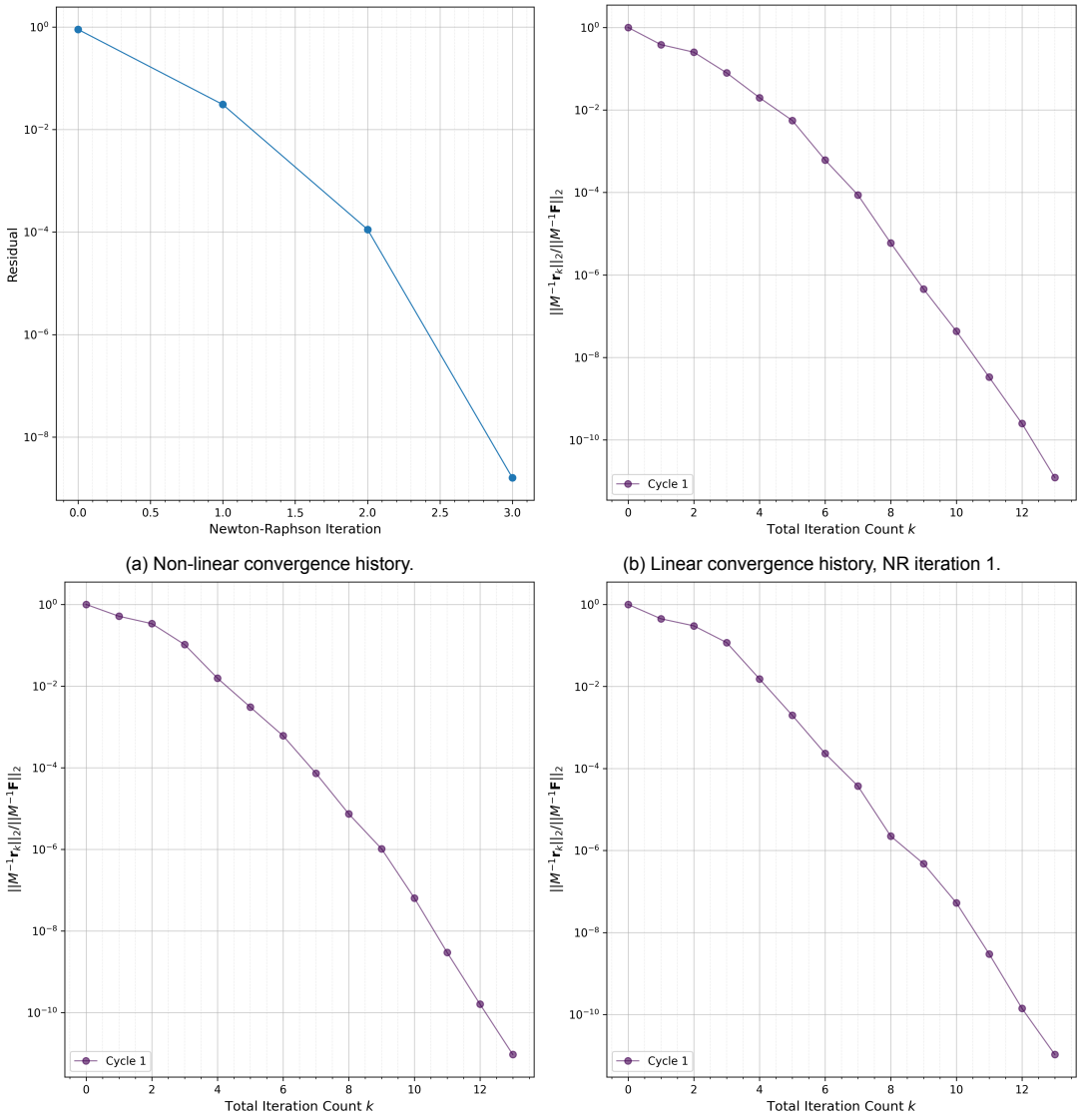
(d) Linear convergence history, NR iteration 3.

Figure 9.15: Convergence history of case30, preconditioned with a basic AS preconditioner for 4 subdomains with ParMETIS, with 2 level overlap, using restarted GMRES as linear solution method.

The linear and non-linear convergence history of the RAS preconditioned case30 network are given in Figure 9.16. The preconditioner was constructed using the ParMETIS algorithm with 4 subdomains and a 2-level overlap. The preconditioned system converged after 3 iteration. Comparing the residuals and number of iterations of the restricted to the basic AS variant, one can see that the residual per iteration is the same, but that RAS required fewer GMRES iterations to converge. In the case of the restricted preconditioner, the linear convergence is also super-linear, but no restarts occurred.

Table 9.9: The non-linear convergence history of the RAS preconditioned case30 network, for 4 subdomains and 2-level overlap. Iteration 0 refers to the initial residual.

NR iteration	Residual	GMRES iterations
0	$8.980 \cdot 10^{-1}$	-
1	$3.073 \cdot 10^{-2}$	13
2	$1.123 \cdot 10^{-4}$	13
3	$1.603 \cdot 10^{-9}$	13



(a) Non-linear convergence history.

(b) Linear convergence history, NR iteration 1.

(c) Linear convergence history, NR iteration 2.

(d) Linear convergence history, NR iteration 3.

Figure 9.16: Convergence history of case30, preconditioned with a RAS preconditioner for 4 subdomains with ParMETIS, with 2 level overlap, using restarted GMRES as linear solution method.

9.2.2. Effect of the Partition

The AS preconditioner construction is associated with two main hyperparameters, the number of subdomains and the level of overlap. Additionally, the employed partitioning method impacts the way in which the subdomains are constructed and which elements are added as overlap. These hyperparameters affect the performance and effectiveness of the AS preconditioner. From theory, it is known that in general an increase of the overlap results in a more effective AS preconditioner that is more costly to compute. Similarly, a decrease in the number of subdomains results in a more effective AS preconditioner that is more costly to compute. This section considers the effect of varying the overlap level, number of subdomain and partitioning algorithm on the AS preconditioner performance applied to small-scale electricity networks.

Level of Overlap

The effect of the level of overlap is investigated by considering the case89pegase test network data. The ParMETIS algorithm is used to partition the network into 5 subdomains. The basic AS preconditioner is applied, varying the overlap level from 1 to 3. Higher overlap levels were not considered for this test case, since an overlap of level 4 or higher results in at least one subdomain being exactly the

entire network. Table 9.10 contains the condition number, largest and smallest singular value, number of linear iterations per NR iteration and the relative size of the sub-system Jacobian for each subdomain for all considered overlap levels. The original condition number of the case89pegase network evaluated at $\mathbf{x}_0^e$ is $5.88 \cdot 10^3$. For all levels of overlap, 4 non-linear iterations were required to converge to the preset tolerance. The non-linear residual history is similar for each overlap level. From the data in the table, one can see that an increase in the level of overlap results in a smaller condition number This indicates a higher quality preconditioner for a higher overlap level. Note that the largest singular value also increases. This is in line with the earlier observation that for the basic AS preconditioner, the spectrum is affected by the number of times a node occurs in a subdomain. With a higher level of overlap, it is more likely that a node is included in almost all subdomains. A higher overlap level also decreases the number of linear iterations required during each NR iteration. The relative sizes of the sub-system Jacobians are also included in the table. In the case of a 3-level overlap, the largest subdomain consists of 97.6% of the global Jacobian. This local subdomain solve comes therefore very close to the cost required to solve the entire global system directly and is computationally more expensive than when a lower overlap level is used.

Table 9.10: Spectral properties and linear solver performance for the case89pegase network, using a basic AS preconditioner, constructed with a ParMETIS partition of 5 subdomains and varying levels of overlap. Bold values in the linear iterations column indicate a restart.

Overlap	Cond. number	Largest SV	Smallest SV	Linear iterations	Sub-system size
1-level	$3.71 \cdot 10^1$	$4.03 \cdot 10^0$	$1.09 \cdot 10^{-1}$	25, **27**, **27**, **27**	30.9% - 49.1%
2-level	$2.88 \cdot 10^1$	$6.19 \cdot 10^0$	$2.15 \cdot 10^{-1}$	23, **25**, **27**, **26**	52.7% - 79.4%
3-level	$1.61 \cdot 10^1$	$7.56 \cdot 10^0$	$4.70 \cdot 10^{-1}$	18, 19, **22**, 19	76.4% - 97.6%

Table 9.11 contains the condition number, extreme singular values, number of linear iterations and sub-system size when the RAS preconditioner is applied to the case89pegase dataset, with 5 subdomains and varying levels of overlap. Compared to Table 9.10, only the AS type is changed. In general, the use of the RAS type results in a smaller condition number for all considered levels of overlap. Additionally, in all three cases less linear iterations were required per NR iteration to converge for the same level of overlap, compared to the basic AS case. The same effects of increasing the overlap level is observed. A higher level of overlap results in a smaller condition number, less linear iterations required for convergence and a higher relative sub-system that is solved directly during the preconditioner construction, indicating a higher computational cost.

Table 9.11: Spectral properties and linear solver performance for the case89pegase network, using a RAS preconditioner, constructed with a ParMETIS partition of 5 subdomains and varying levels of overlap. Bold values in the linear iterations column indicate a restart.

Overlap	Cond. number	Largest SV	Smallest SV	Linear iterations	Sub-system size
1-level	$3.13 \cdot 10^1$	$2.58 \cdot 10^0$	$8.25 \cdot 10^{-1}$	**21**, **22**, **24**, **26**	30.9% - 49.1%
2-level	$1.84 \cdot 10^1$	$2.48 \cdot 10^0$	$1.35 \cdot 10^{-1}$	**14**, **18**, **15**, **15**	52.7% - 79.4%
3-level	$1.05 \cdot 10^1$	$2.73 \cdot 10^0$	$2.60 \cdot 10^{-1}$	**11**, **12**, **13**, **16**	76.4% - 97.6%

Number of Subdomains

The effect of the number of subdomains is investigated by considering the case89pegase test network data. The ParMETIS algorithm is used to partition the network into various numbers of subdomains, ranging between 2 and 16. The case of 16 subdomains is considered since it is the maximum number of subdomains for this dataset according to the ParMETIS algorithm. The RAS preconditioner is applied, using a 2-level overlap. Table 9.12 contains the condition number, extreme singular values, number of linear iterations and the relative sub-system sizes for all considered number of subdomains. This relative sub-system size is computed as the percentage of the global Jacobian size included in the subdomain. For all number of subdomains, 4 NR iterations were required for convergence to the preset tolerance. Overall, the condition number tends to increase when the number of subdomains increases, although not monotonically. A smaller condition number indicates a better performing preconditioner. This is also visible in the number of linear iterations per NR iteration, that increase with the number

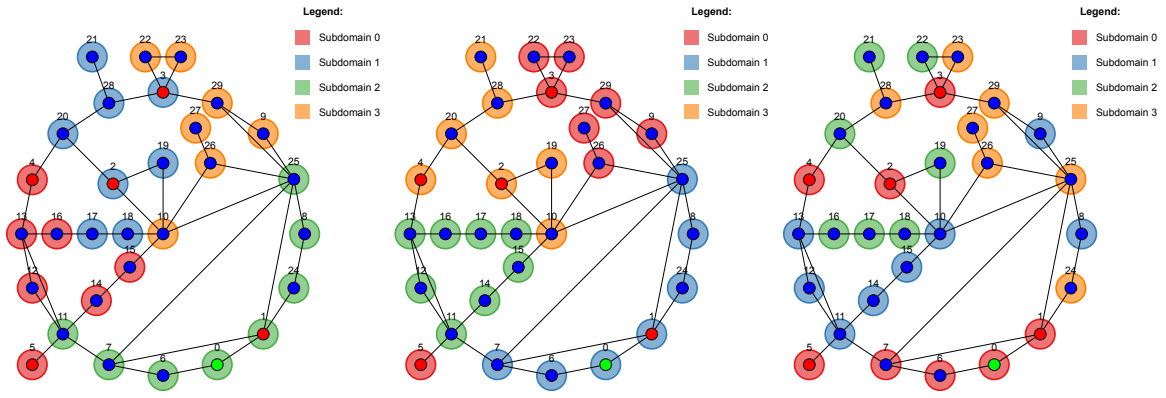
of subdomains. Note that when the network graph is partitioned into more subdomains, there is more variation possible in connectivity of the subdomains. This can result in a wide range of relative sub-system Jacobian sizes. The connectivity of the subdomains not only determines the cost of constructing the preconditioner, but also increases the preconditioners' ability to capture global behaviour. This connectivity effect can therefore overshadow the effect of changing the number of subdomains on the quality of the preconditioner.

Table 9.12: Spectral properties and linear solver performance for the case89pegase network, using a RAS preconditioner with a 2-level overlap, constructed with a ParMETIS partition of a varying number of subdomains. Bold values in the linear iterations column indicate a restart.

Subdomains	Cond. number	Largest SV	Smallest SV	Linear its.	Sub-syst. size
2	$1.04 \cdot 10^{1}$	$2.72 \cdot 10^{0}$	$2.62 \cdot 10^{-1}$	9, 9, 9, 9	90.3% - 95.2%
3	$2.00 \cdot 10^{1}$	$2.97 \cdot 10^{0}$	$1.48 \cdot 10^{-1}$	**15**, **16**, **16**, **17**	64.2% - 82.4%
4	$2.39 \cdot 10^{1}$	$2.61 \cdot 10^{0}$	$1.09 \cdot 10^{-1}$	**16**, **15**, **16**, **16**	71.5% - 79.4%
5	$1.84 \cdot 10^{1}$	$2.48 \cdot 10^{0}$	$1.35 \cdot 10^{-1}$	**14**, **18**, **15**, **15**	52.7% - 79.4%
6	$1.73 \cdot 10^{1}$	$2.49 \cdot 10^{0}$	$1.44 \cdot 10^{-1}$	**16**, **16**, **17**, **17**	39.4% - 73.3%
16	$7.35 \cdot 10^{1}$	$3.24 \cdot 10^{0}$	$4.41 \cdot 10^{-2}$	**22**, **31**, **28**, **23**	24.8% - 75.2%

Partitioning Method

The effect of the partitioning method is investigated by considering the case30 test network data. This dataset is considered as it is small enough the visualise a partition, but large enough that variations in partition can occur. When no preconditioner is applied, the case30 network initial Jacobian has a condition number of $4.93 \cdot 10^{2}$, and requires 53 linear iterations per NR iteration to converge. The ParMETIS, CHACO and average algorithms are used to partition the case30 network into 4 subdomains, given in Figure 9.17. The ParMETIS and CHACO algorithms are both multi-level partitioning algorithms, and differ in the methods used in the coarsening, partitioning and refinement phases. The average algorithm does not take the network topology into account. It computes the average number of nodes required per subdomain to distribute the total number of nodes evenly among all subdomains. Then the nodes are added to the subdomain based on their number, such that the first n nodes are included in the first subdomain and so on. The numbering of the nodes is such that the slack node is the first node, then come all PV nodes, and finally all PQ nodes.



(a) ParMETIS (b) CHACO (c) Average

Figure 9.17: Network graph of the case30 dataset, partitioned into 4 subdomains using different algorithms. A red node represents a PV node, a blue node represents a PQ node and the slack node is indicated with a green colour.

From Figure 9.17, one can see that the ParMETIS and CHACO partitioning algorithms both contain subdomains with disconnected components. For example, subdomain 3 in the ParMETIS partition consists of three components that are not directly connected. Such a subdomain is more likely to have a higher number of adjacent nodes. When an overlap is added during the AS construction, such a disconnected subdomain can result in larger sub-system Jacobians than connected subdomains. The disconnected components can become connected when overlap is added.

The RAS preconditioner is constructed using the three partitioning methods mentioned above, using 1- and 2-level overlaps. Table 9.13 contains the condition number, number of linear iterations and relative sub-system sizes of the considered cases. Additionally, Table 9.14 shows the condition number, linear iteration history and sub-system sizes for the case89pegase dataset. For all three partitioning algorithms, the application of the RAS preconditioner improved the condition number compared to no preconditioner application, as well as decreased the number of linear iterations required per NR iteration. The partitioning algorithm affects the relative sub-system Jacobian size. The average method, which contains the most disconnected components, results in the largest range in relative sub-system Jacobian size for both considered overlap levels, reaching 92.5%.

Table 9.13: Spectral properties and linear solver performance for the case30 network, using a RAS preconditioner constructed with a partition into 4 subdomains using the ParMETIS, CHACO and average partitioning algorithms, with a 1- and 2-level overlap. Bold values in the linear iterations column indicate a restart.

Algorithm	Overlap	Condition number	Linear iterations	Sub-system size
ParMETIS	1-Level	$2.47 \cdot 10^1$	15, 15, 15	37.8% - 50.9%
	2-Level	$8.85 \cdot 10^0$	13, 13, 13	57.9% - 77.4%
CHACO	1-Level	$1.73 \cdot 10^1$	**16**, **16**, **16**	39.6% - 43.4%
	2-Level	$1.05 \cdot 10^1$	12, 12, 12	64.2% - 90.1%
Average	1-Level	$2.36 \cdot 10^1$	**18**, **18**, **18**	47.2% - 66.0%
	2-Level	$5.94 \cdot 10^0$	11, **13**, **13**	69.8% - 92.5%

Table 9.14: Spectral properties and linear solver performance for the case89pegase network, using a RAS preconditioner constructed with a partition into 5 subdomains using the ParMETIS, CHACO and average partitioning algorithms, with a 1- and 2-level overlap. Bold values in the linear iterations column indicate a restart.

Algorithm	Overlap	Condition number	Linear iterations	Sub-system size
ParMETIS	1-Level	$3.13 \cdot 10^1$	**18, 20, 21, 22**	36.3% - 52.7%
	2-Level	$2.39 \cdot 10^1$	**16, 15, 17, 16**	71.5% - 79.4%
CHACO	1-Level	$2.94 \cdot 10^1$	**20, 21, 22, 22**	33.3% - 51.5%
	2-Level	$2.22 \cdot 10^1$	**14, 14, 15, 16**	52.1% - 80.6%
Average	1-Level	$6.50 \cdot 10^1$	**33, 42, 42, 39**	63.6% - 70.9%
	2-Level	$1.43 \cdot 10^1$	**13, 13, 14, 14**	92.1% - 96.4%

9.2.3. AS Types: Basic, Interpolated and Restricted

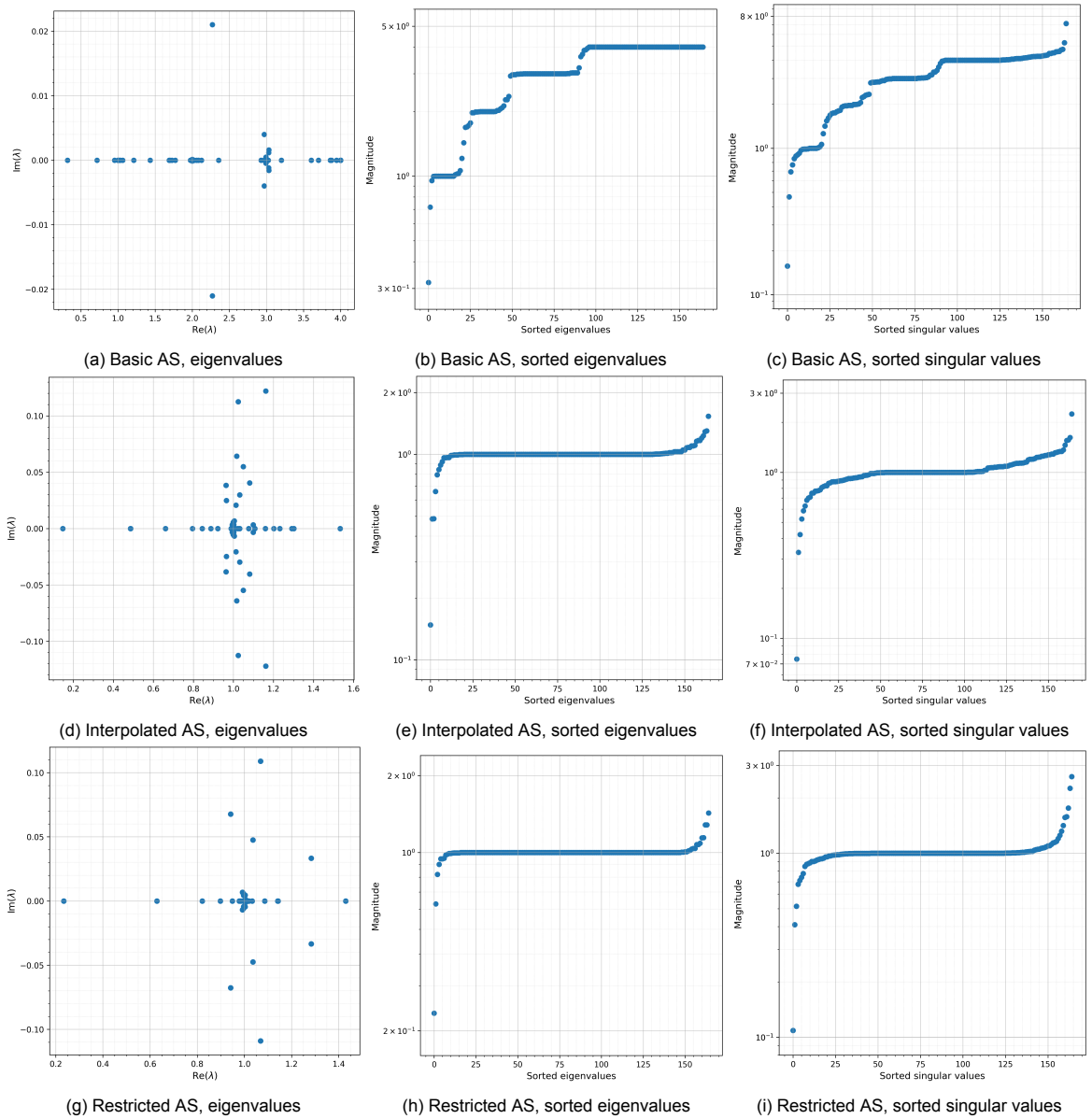
The three types of AS preconditioner considered in this research are basic, interpolated and restricted, see Section 6.2.3. They differ in the prolongation step, where the local updates are transferred back to the global basic. The application of the RAS preconditioner variant has been investigated when applied to large power flow systems [1], showing good scalability and speed-up. This section investigates the performance of the three different variants of the AS preconditioner. The effect on the preconditioned Jacobian spectrum, convergence history and error propagation are discussed.

Spectrum of the Jacobian

Figure 9.18 gives an overview of the eigenvalues and singular values of the preconditioned Jacobian of the case89pegase network, comparing the basic AS, interpolated AS and RAS types. The condition numbers, extreme singular values, linear iteration counts and final residuals are summarised in Table 9.15. For all types, the eigenvalues are complex and have positive real part. The sorted eigenvalues of the basic AS case shows clusters of eigenvalues at 1, 2, 3 and 4. The number of eigenvalues that are exactly $n \in \{1, 2, 3, 4\}$ is smaller than the number of elements included in n subdomains. This pattern is also present for the singular values, where multiple singular values are exactly 1, 2, 3 or 4, although there are less of these integer-valued singular values in total. When the interpolated and RAS types are used, all integer-values eigenvalues and singular values are 1. In these cases, no network element is updated more than once. The extreme singular values of the basic AS preconditioned system are slightly more extreme than those of the interpolated and restricted cases.

Table 9.15: Spectral properties and linear solver performance for the case89pegase network, using a basic, interpolated and RAS preconditioner, constructed with a ParMETIS partition of 4 subdomains and a 2-level overlap. Bold values in the linear iterations column indicate a restart.

AS type	Cond. number	Largest SV	Smallest SV	Linear iterations	Final residual
Basic	$4.55 \cdot 10^{1}$	$7.13 \cdot 10^{0}$	$1.57 \cdot 10^{-1}$	**24**, 23, **25**, 23	$2.726101 \cdot 10^{-9}$
Interpolated	$3.00 \cdot 10^{1}$	$2.25 \cdot 10^{0}$	$7.49 \cdot 10^{-2}$	**18, 21, 22, 22**	$2.726088 \cdot 10^{-9}$
Restricted	$2.39 \cdot 10^{1}$	$2.61 \cdot 10^{0}$	$1.09 \cdot 10^{-1}$	**16, 15, 17, 16**	$2.726049 \cdot 10^{-9}$



(a) Basic AS, eigenvalues (b) Basic AS, sorted eigenvalues (c) Basic AS, sorted singular values

(d) Interpolated AS, eigenvalues (e) Interpolated AS, sorted eigenvalues (f) Interpolated AS, sorted singular values

(g) Restricted AS, eigenvalues (h) Restricted AS, sorted eigenvalues (i) Restricted AS, sorted singular values

Figure 9.18: Eigenvalues and singular values of the preconditioned global Jacobians of case89pegase, with a basic, interpolated and RAS preconditioner for 4 subdomains with ParMETIS and 2-level overlap, evaluated at initial iterate $\mathbf{x}_0^g$.

Convergence history

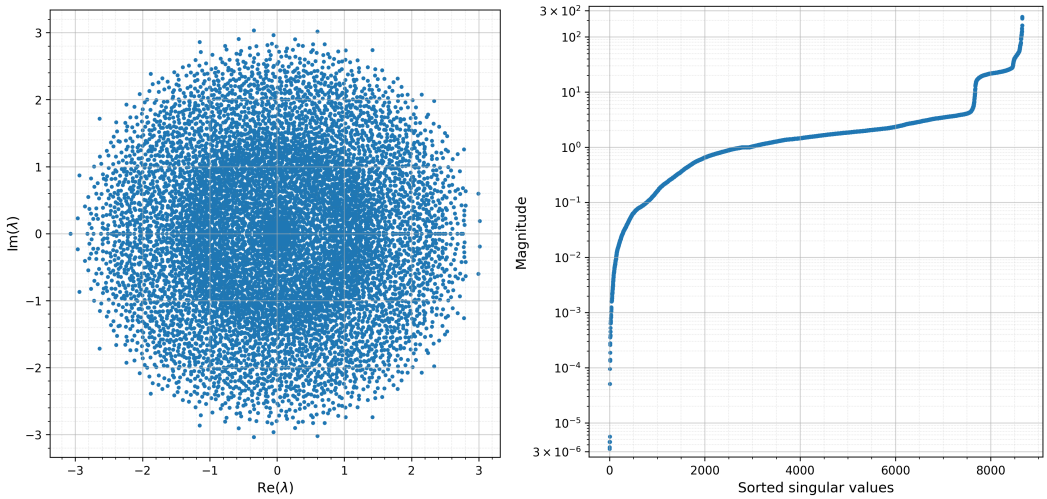
Table 9.15 contains the number of linear iterations per NR iteration, as well as the final residual after solving the steady-state load flow equations of the case89pegase dataset, using basic, interpolated and RAS preconditioners. The RAS has the smallest conditioner number, as well as the least number of linear iterations required per NR iteration. The differences in final residual between the AS types is very small.

9.3. Large-Scale Gas and Electricity Networks

The performance and application of the AS preconditioner for small-scale gas and electricity networks has been presented and discussed in the previous two sections, Section 9.1 and Section 9.2, respectively. The aim of this research is to apply the AS preconditioner in the solution process of load flow analysis associated with large-scale and coupled networks. The scalability of the preconditioned iterative linear solvers is therefore important to consider.

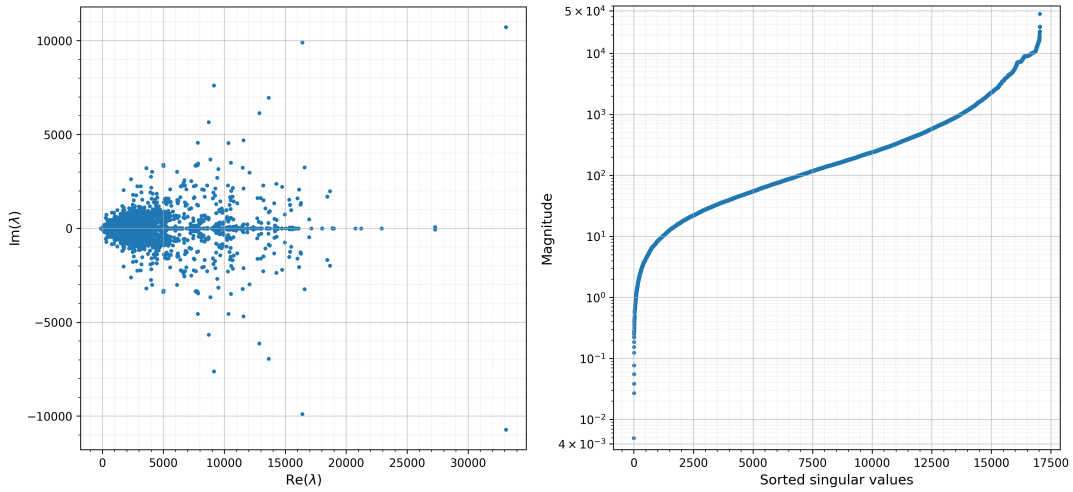
This section analyses and discusses the performance of the AS preconditioner applied to large-scale gas and electricity networks, focussing on scalability and a comparison to small-scale networks. First, the general effect is discussed, considering convergence, cost and accuracy. Additionally, the effect of partition related parameters and the different AS types is discussed. The section concludes by investigating the effect of reusing a previous preconditioner during a later NR iteration, as was also done in research into incomplete LU (ILU) preconditioner for large power flow networks [23]. Note that for the large-scale networks the computation of the eigenvalues, singular values and the condition number is computationally expensive. Therefore, these quantities do not play such a central role in the AS preconditioner performance analysis compared to small-scale networks. Instead, there is more focus on the linear convergence history.

The largest gas and electricity networks considered in this research are the GasLib-4197 and case9241pegase datasets, respectively, see Section 8.1. The spread of the eigenvalues and singular values of the original initial GasLib-4197 dataset are given in Figure 9.19. Similarly to small-scale gas networks, the eigenvalues are centred around zero with a (relatively small) radius of 3. Such a spread around zero is undesirable when using iterative Krylov methods. The spread of the eigenvalues and singular of the original initial case9241pegase dataset are given in Figure 9.20. All eigenvalues have positive real part and show a fan-like structure emerging from zero. The magnitudes of the largest eigenvalues are more extreme than the largest eigenvalues of the smaller PandaPower datasets.



(a) Eigenvalues (b) Singular values

Figure 9.19: Eigenvalues and singular values of the initial Jacobian of the GasLib-4197 gas network dataset.



(a) Eigenvalues (b) Singular values

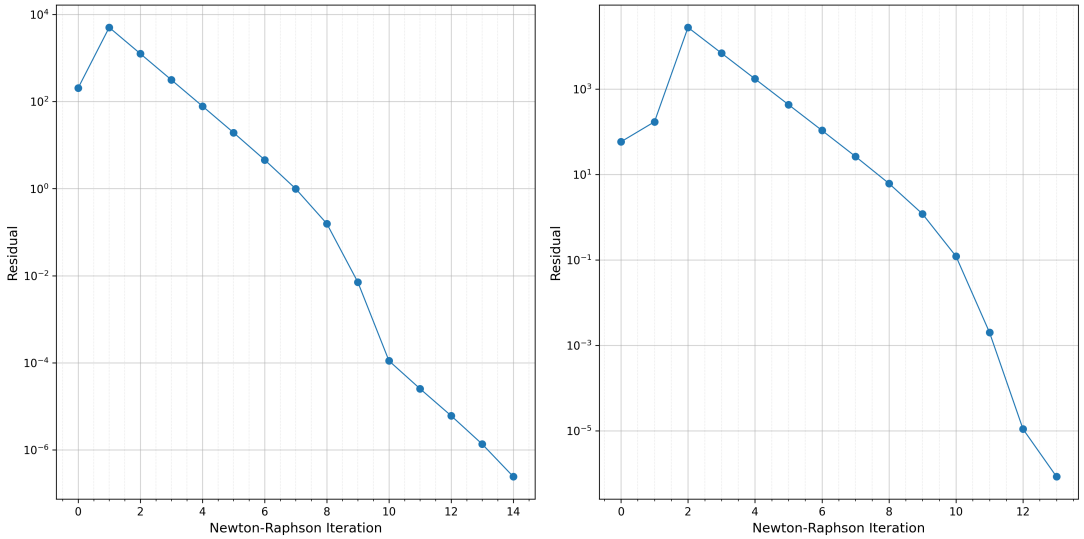
Figure 9.20: Eigenvalues and singular values of the initial Jacobian of the case9241pegase electricity network dataset.

9.3.1. General Effect of AS Preconditioning

The convergence history of small-scale gas and electricity networks show super-linear convergence, for the non-linear as well as linear (GMRES) iterations. The convergence of the linear system per NR iteration for small-scale gas networks has a more prominent super-linear pattern than for small-scale electrical networks.

Non-Linear Convergence History: Gas Networks

Figure 9.21 shows the non-linear convergence history and residuals when solving the steady-state load flow equations of the GasLib-582 and GasLib-4197 datasets, where a direct solve is used in each NR iteration. The GasLib-582 dataset requires 14 NR iterations, and the GasLib-4197 requires 13. The final residuals and number of NR iterations required are given in Table 9.16. The steady-state load flow equations of the GasLib-2607 dataset cannot be solved using a direct solve per NR iteration, due to a singular Jacobian matrix during the second NR iteration. A noticeable difference in the non-linear residuals compared to the smaller GasLib datasets is the increase in residual during the first iterations, see Figure 9.21. After the initial increase, there is super-linear convergence.



(a) GasLib-582 (b) GasLib-4197

Figure 9.21: Non-linear residual history of the GasLib-582 and GasLib-4197 datasets, using a direct solve per NR iteration.

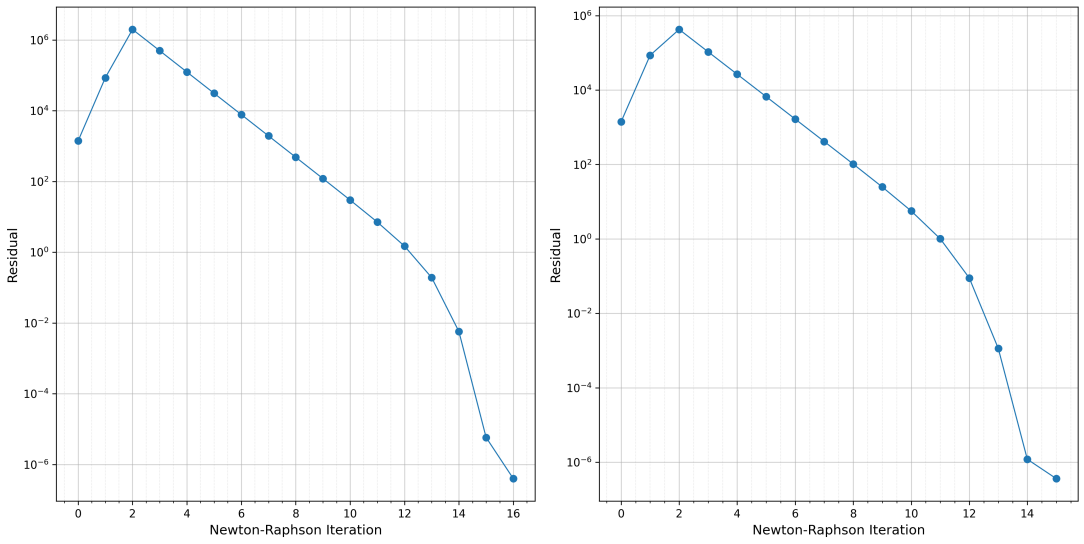
Table 9.16: Number of NR iterations required for the large-scale GasLib datasets to converge to the pre-set tolerance using a direct solve per NR iteration and RAS preconditioner GMRES(m) with 4 subdomains from the ParMETIS algorithm and a 3-level overlap, and the final residuals. The GasLib-2607 dataset cannot be solved using a direct solve per NR iteration.

Network	NR iterations	Final residual	
		Direct solve	RAS + GMRES(m)
GasLib-582	14	$2.428902 \cdot 10^{-7}$	$2.428902 \cdot 10^{-7}$
GasLib-2607	19	-	$2.608875 \cdot 10^{-7}$
GasLib-4197	13	$8.509677 \cdot 10^{-7}$	$8.509604 \cdot 10^{-7}$

When the AS preconditioner is used during each NR iteration, the same non-linear residuals and patterns are observed compared to using a direct solve for the GasLib-582 and GasLib-4197 networks. The final residuals do not differ significantly with the final residual when a direct linear solve is done, see Table 9.16. The non-linear residual history is the same as shown in Figure 9.21.

The non-linear residual history of the large-scale GasLib datasets is different, compared to the small-scale datasets, due to the initial increase in non-linear residual observed for the large-scale datasets. After this initial increase, small-scale as well as large-scale networks show super-linear NR convergence.

As indicated in Table 9.16, the GasLib-2607 dataset cannot be solved using a direct linear solve, due to a singular Jacobian matrix. The non-linear residual does converge when GMRES is used as a linear solver, preconditioned with AS constructed using a certain combination of number of subdomains and overlap-levels. For example, when a RAS preconditioner is used, where the partition into 4 subdomains is made with ParMETIS, and a 2-level overlap is applied, there are 16 NR iterations required to converge, see Figure 9.22a. The same pattern can be observed as for the GasLib-582 and GasLib-4197 datasets, Figure 9.21, where there is an initial increase of the residual, after which the residual converges. When the partition is made consisting of 2 subdomains and a 1-level overlap, see Figure 9.22b, the non-linear convergence history shows the same shape but required 15 iterations, which is fewer NR iterations compared to the 4 subdomains and 2-level overlap case. When the RAS preconditioner is constructed using 4 subdomains and a 3-level overlap, the number of NR iterations increases to 19, as indicated in Table 9.16. The NR convergence of this dataset is thus more sensitive to the preconditioner configuration than the other considered large-scale GasLib datasets.



(a) 4 subdomains, 2-level overlap

(b) 2 subdomains, 1-level overlap

Figure 9.22: Non-linear residual history of the GasLib-2607 dataset, using a RAS preconditioner, constructed using a different number of subdomains and overlap.

Linear Convergence History: Gas Networks

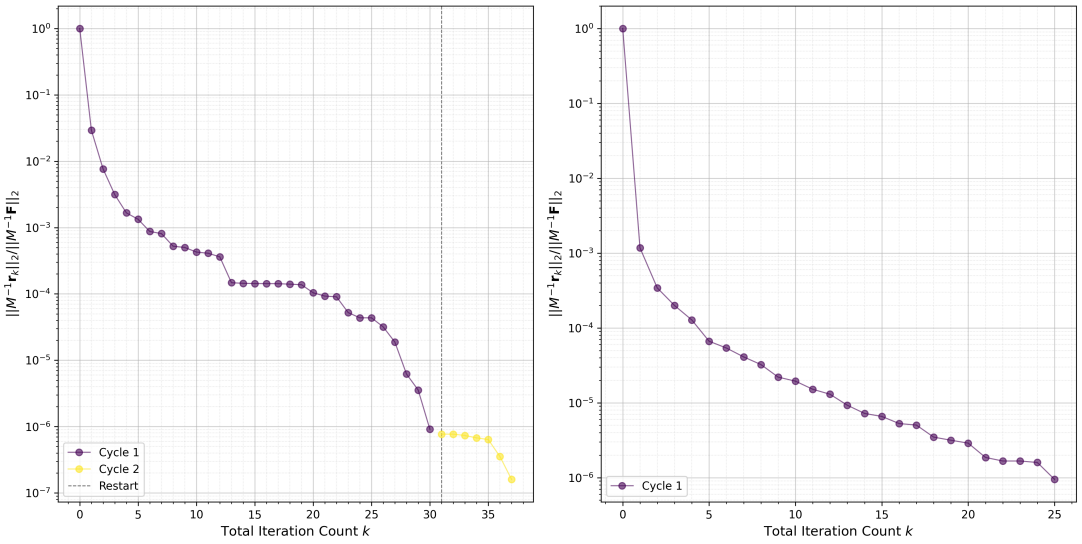
Table 9.17 contains the linear convergence history of the GasLib-4197 dataset, when a RAS preconditioned GMRES is applied during the linear solution steps, with various numbers of subdomains and levels of overlap. The effect of varying the overlap and the number of subdomains in discussed in Section 9.3.3.

Table 9.17: Linear convergence and final residual observed for the GasLib-4197 network, using a RAS preconditioner, constructed using various subdomain sizes and overlap levels with the ParMETIS algorithm. Bold values in the linear iterations column indicate a restart.

Subdomains	Overlap	Linear iterations	Final residual
2	1-level	14, 14, 13, 14, 10, 12, 10, 10, **21**, 7, **19**, **21**, **21**	$8.509152 \cdot 10^{-7}$
4	2-level	**37**, 30, 30, 25, 26, 25, **48**, **70**, **62**, **19**, **60**, **71**, **40**	$8.509359 \cdot 10^{-7}$
6	3-level	**54, 65**, 42, 26, 35, **47**, 27, **57**, **71**, **30**, **65**, **80**, **77**	$8.509676 \cdot 10^{-7}$

The linear system convergence behaviour during each NR iteration also shows differences for large-scale GasLib networks compared to the small-scale networks. The linear convergence history of the first and sixth NR iteration of the GasLib-4197 are shown in Figure 9.23. As the linear solver, RAS preconditioned restarted GMRES is used, where the preconditioner is constructed using 4 subdomains and a 2-level overlap. During the first iteration, a stagnation-acceleration pattern is observed. The residual first drops, showing sub-linear convergence, after which there is a stagnation phase. After several iterations where the relative preconditioned residual does not change that much, this residual decreases super-linearly. This pattern is often observed in Krylov subspace methods. During the initial sub-linear phase, the dominant eigenvalues are eliminated, resulting in a larger decrease in the residual. During the stagnation phase, the residual curve flattens out. The GMRES algorithm cannot effectively span the required space. After several iterations, the Krylov subspace can sufficiently eliminate these components, resulting in super-linear convergence. During a later NR iteration, such as the sixth iteration shown in Figure 9.23b, only the sub-linear phase is observed. Although not shown in the figure, it could be that when the tolerance is set tighter, the same stagnation-acceleration pattern would be observed, and that during this iteration the preconditioned relative residual has sufficiently decreased during the initial phase already.

The number of linear iterations required per NR iteration when using a AS preconditioner for large-scale gas has increased, compared to small-scale networks. The GasLib-4197 dataset contains about 100 times as many nodes as the GasLib-40 dataset. The average number of linear iterations per NR iteration is larger, but still of the same order of magnitude.



(a) NR iteration 1 (b) NR iteration 6

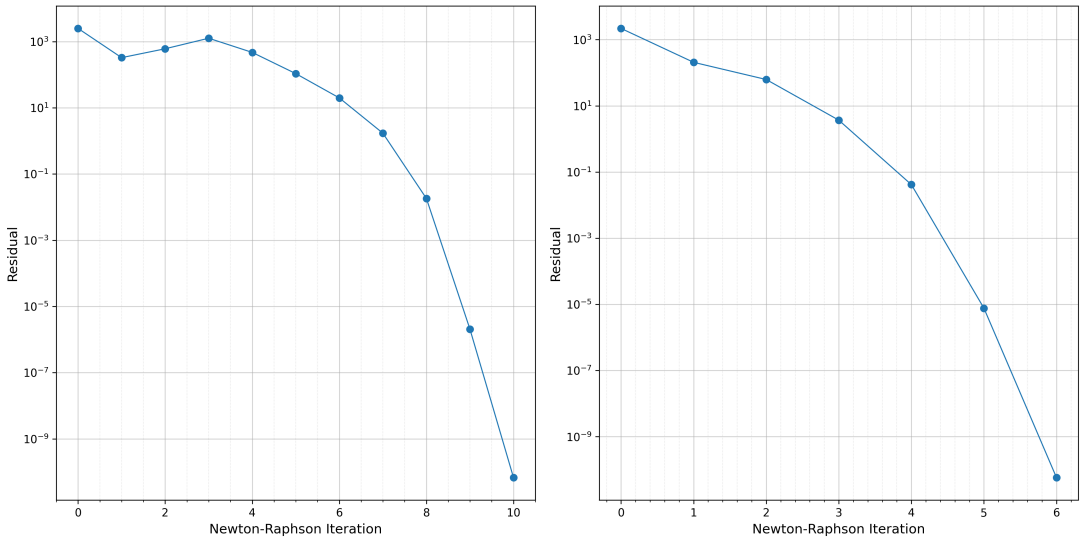
Figure 9.23: Linear residual history of the GasLib-4197 dataset during the 1st and 6th NR iteration, using a RAS preconditioner, constructing using a 4 subdomains with the ParMETIS algorithm and 2-level overlap.

Non-Linear Convergence History: Electricity Networks

Table 9.18 shows the number of NR iterations required for the non-linear residual of the large-scale PandaPower datasets to become smaller than the tolerance of $1.0 \cdot 10^{-6}$ as well as the final residual when using a direct solve as the linear solve per NR iteration. The non-linear residual history of the case6515rte and case9241pegase networks are given in Figure 9.24. The non-linear convergence history of the other large-scale datasets given in Table 9.18 follow a pattern similar to the super-linear convergence observed in Figure 9.24b. The non-linear residual of the case6515rte is the only dataset where a pattern (slightly) similar to the large-scale GasLib datasets' non-linear history is observed, such that the residual increases before it shows super-linear convergence. The case6515rte dataset requires roughly twice as many NR iterations compared to the other large-scale PandaPower networks.

Table 9.18: Number of NR iterations required for the large-scale PandaPower datasets to converge to the pre-set tolerance using a direct solve per NR iteration, and the final residuals.

Network	NR iterations	Final residual	
		Direct solve	RAS + GMRES(m)
case1888rte	6	$6.015587 \cdot 10^{-11}$	$6.082225 \cdot 10^{-11}$
case2848rte	6	$5.621872 \cdot 10^{-11}$	$5.972308 \cdot 10^{-11}$
case3120sp	5	$9.958756 \cdot 10^{-7}$	$9.958789 \cdot 10^{-7}$
case6515rte	10	$6.744816 \cdot 10^{-11}$	$6.699047 \cdot 10^{-11}$
case9241pegase	6	$5.769165 \cdot 10^{-11}$	$5.748121 \cdot 10^{-11}$



(a) case6515rte (b) case9241pegase

Figure 9.24: Non-linear residual history of the case6515rte and case9241pegase datasets, using a direct solve per NR iteration.

When AS preconditioned GMRES is applied during each NR iteration, the same non-linear residuals and patterns are observed compared to using a direct solve, see Table 9.18. The non-linear convergence history of large-scale PandaPower networks when using a RAS preconditioner is similar to the small-scale networks, showing super-linear convergence.

Linear Convergence History: Electricity Networks

Table 9.19 summarizes the linear convergence history and final residual when the RAS preconditioner is applied during the linear solves of the case9241pegase dataset, where the AS preconditioner is constructed such that the network is partitioned into varying numbers of subdomains using the ParMETIS algorithm, and varying levels of overlap. The differences in final residuals, also compared to the case of a direct solve, are not significant. The non-linear residual history is the same as shown in Figure 9.24b. The effect of varying the overlap and number of subdomains is discussed further in Section 9.3.3.

Table 9.19: Linear convergence and final residual observed for the case9241pegase network, using a RAS preconditioner, constructed using various subdomain sizes and overlap levels with the ParMETIS algorithm. Bold values in the linear iterations column indicate a restart.

Subdomains	Overlap	Linear iterations	Final residual
2	1-level	**24, 27, 26, 25, 30, 27**	$5.705376 \cdot 10^{-11}$
4	2-level	**39, 44, 39, 45, 39, 41**	$5.748121 \cdot 10^{-11}$
6	3-level	34, **36, 37, 35, 36, 37**	$5.556804 \cdot 10^{-11}$

The linear convergence history of the large-scale PandaPower networks when using a RAS preconditioner is similar to the small-scale PandaPower networks, showing a super-linear pattern. Additionally, there is at least one restart during most of the linear iterations. After the restart, there are usually only a few linear iterations required before the restarted GMRES solver terminates. The large-scale PandaPower networks consist of about 20 to 300 times as many nodes compared to the considered small-scale datasets (case30 and case89pegase). Despite this size increase, the average number of linear iterations per NR iteration is still of the same order of magnitude.

9.3.2. Effect of Network Elements

Section 9.1.2 describes the effects that the gas network elements resistors and compressors can have on the AS preconditioner construction and performance, particularly observed in the GasLib-24 and GasLib-135 datasets. The described effects and issues apply to all gas networks, but they do not

occur in all datasets. An additional effect of network elements on the AS preconditioner performance is observed in the GasLib-582 dataset.

Zero link flow
The solution of the steady-state load flow of the GasLib-582 dataset contains several links with zero gas mass flow. This implies that there is no gas flowing through this element and that the nodal pressures of the nodes connected to the link are equal.

During the construction of the AS preconditioner, several sub-system Jacobians are constructed. In the case of zero gas mass flow across link elements, a singular sub-system Jacobian can occur. The cause of the singularity is described below, after which it is illustrated with a small example.

The mathematical explanation for the singularity is the occurrence of linearly dependent rows and columns. The sub-system Jacobian corresponding to a subdomain of a partitioned gas network has the following structure

$$J_{F^g} = \begin{bmatrix} \frac{\partial \mathbf{F}^q}{\partial \mathbf{q}} & \frac{\partial \mathbf{F}^q}{\partial \mathbf{p}} \\ \frac{\partial \mathbf{F}^L}{\partial \mathbf{q}} & \frac{\partial \mathbf{F}^L}{\partial \mathbf{p}} \end{bmatrix},$$

where the components corresponding to the link equations are given by

$$\begin{aligned}
\frac{\partial \mathbf{F}_k^L}{\partial q_k} &= -2(C^g)^{-2}|q_k|\left(f_k + \frac{1}{2}q_k\frac{\partial f_k}{\partial q_k}\right), \\
\frac{\partial \mathbf{F}_k^L}{\partial q_l} &= 0, \\
\frac{\partial \mathbf{F}_k^L}{\partial p_i} &= \begin{cases} 1, & \text{low-pressure,} \\ 2p_i, & \text{high-pressure,} \end{cases} \\
\frac{\partial \mathbf{F}_k^L}{\partial p_j} &= \begin{cases} -1, & \text{low-pressure,} \\ -2p_j, & \text{high-pressure.} \end{cases}
\end{aligned}$$

When the gas mass flow through link k becomes zero, we get $q_k = 0$, resulting in $\partial \mathbf{F}_k^L / \partial q_k = 0$. Therefore, the row in $\partial \mathbf{F}^L / \partial \mathbf{q}$ corresponding to the link equation of link k (the lower left block) consists of only zeros. This alone does not result in a singular matrix, as the entries in lower right block $\partial \mathbf{F}^L / \partial \mathbf{p}$ prevent linear dependence.

When a link is a boundary link, so one node is included in the subdomain and the other is not, there is only one unknown nodal pressure variable within the subdomain. This means that in the lower right block $\partial \mathbf{F}^L / \partial \mathbf{p}$, in the row associated with link k connected to nodes i and j, there is only one entry. This is either $\partial \mathbf{F}_k^L / \partial p_i$ or $\partial \mathbf{F}_k^L / \partial p_j$, but not both, as one node is not included in the subdomain. When there are two such boundary links connected to the same node in the subdomain, the rows associated with both links in $\partial \mathbf{F}^L / \partial \mathbf{p}$ are equal (up to multiplication with a constant). This alone does not result in a singular sub-system Jacobian, as the entries in the lower left block $\partial \mathbf{F}^L / \partial \mathbf{q}$ prevent linear dependence.

When the situations described above are combined, so that there is a subdomain where there are two or more boundary links connected to the same node and that these two boundary links have converged to zero flow across the links, the sub-system Jacobian becomes singular. The rows and columns associated with these boundary links are linearly dependent. During the AS construction, the update of this subdomain can therefore not be computed.

An example of a subdomain with a singular sub-system Jacobian is given in Figure 9.25. It consists of 4 nodes, 2 of which are included in the subdomain, and three low-pressure gas pipes. Other link elements, such as compressors and resistors, also result in singular Jacobians, as only the entry within the Jacobian changes, not the non-zero structure. For simplicity, the example network contains low-pressure gas pipes. The gas mass flows along links q_{32} and q_{24} have become zero, so that there is no gas flowing through these link elements. The subdomain x consists of nodes 1 and 2, as well as links

q_{21}, q_{32} and q_{24}. This example matches the situation described earlier in this section, where there are two zero flow boundary links connected to the same node.

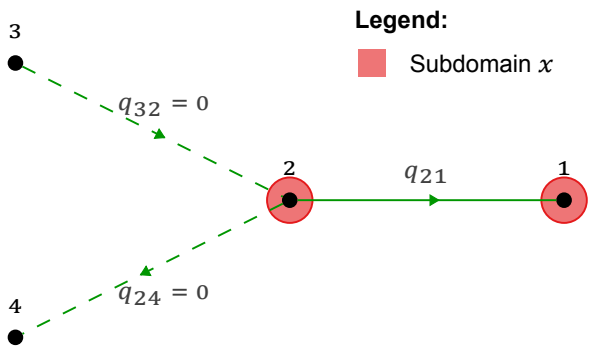


Figure 9.25: Example of a subdomain i with two zero flow boundary links (dashed lines), resulting in a singular sub-system Jacobian.

The sub-system Jacobian matrix associated with subdomain x, evaluated at current iterate $\mathbf{x}_i^g$ is

$$J_x(\mathbf{x}_i^g) = \left[\begin{array}{ccc:cc} 1 & & & & \\ -1 & 1 & -1 & & \\ \hdashline a & & & 1 & -1 \\ & b = 0 & & & 1 \\ & & c = 0 & & -1 \end{array}\right], \tag{9.1}$$

where $a = 2(C^g)^{-2}|q_{21}|\left(f_{21} + \frac{1}{2}q_{21}\frac{\partial f_{21}}{\partial q_{21}}\right)$. The entries b and c are zero due to the zero flow along links q_{32} and q_{24}. Since these links are boundary links, there is only 1 entry in the lower right block of the last two rows. In (9.1), the fourth and fifth rows are linearly dependent, as well as the second and third column. This Jacobian is therefore singular.

The singularity of the sub-system Jacobian can also be interpreted physically. When a link is a boundary link, one connected nodal pressure, say p_i, is unknown, while the other nodal pressure p_j is fixed. Since there is zero flow through these link elements, the link equations imply that the nodal pressure at both nodes must be equal. Since p_j is fixed, p_i has to be updated such that is equals p_j. When there are two of such zero flow links connected to the same node, the nodal pressure p_i has to be updated such that it is exactly equal to two fixed nodal pressures. Only when these are the same, is it possible to determine the updated nodal pressure p_i. When they are not the same, there is no feasible solution, leading to singularity.

In Section 9.1.2 it is described that when two boundary links connected to the same node are both compressors, the corresponding sub-system Jacobian is singular. The mathematical cause of the singularity, the linearly dependent rows and columns, is the same as the situation described in this section. The link equation used to model a compressor does not contain the link flow, automatically resulting in a zero row in the $\partial \mathbf{F}^L / \partial \mathbf{q}$ component. The reason for the singularity is different as it is associated with different network elements, but their effect on the sub-system Jacobian is similar.

A link without zero gas mass flow through the link does not result in a singular Jacobian when solving the steady-state load flow problem directly. The singularity can only occur in sub-system Jacobians of the subdomains. It is not possible to know beforehand which link elements will converge to zero flow, so finding a partition that is guaranteed to have only non-singular Jacobians will not solve the issue. It is however possible to change the subdomains per NR iteration. When a link shows a very small or zero gas flow, this link can be removed from the subdomain. The sub-system Jacobian is no longer singular and the AS preconditioner construction can continue. However, the link flow can no longer be updated when it is removed.

9.3.3. Effect of the Partition

The effect of partition-related hyperparameters was investigated and discussed for small-scale gas and electricity networks in Section 9.1.3 and Section 9.2.2, respectively. The results presented show that an increase in overlap-level as well as a decrease in number of subdomains resulted in a more effective RAS preconditioner, such that the number of linear iterations decreased. However, these situations are associated with higher computational costs due to larger relative sub-system sizes of the local problems. Additionally, for small-scale networks the usage of a multi-level partitioning method is preferred over the 'average' method that does not consider the graph network structure. There were no significant difference between the different employed multi-level algorithms. Therefore, the ParMETIS multi-level partitioning algorithm will be used for large-scale networks.

Level of overlap

For small-scale gas networks, increasing the level of overlap results in a decrease in the condition number of the initial preconditioned Jacobian as well as the number of linear iterations per NR iteration. A higher overlap level thus results in a more effective preconditioner. On the other hand, increasing the overlap level also increases the size of the local subdomain problems to be solved, which increases the computational cost of constructing the preconditioner. To investigate whether the same effects of changing the overlap level hold for large-scale gas networks, the GasLib-4197 dataset is considered. A partition into 4 subdomains of the network graph is made using the ParMETIS algorithm. The RAS preconditioner is then applied to the restarted GMRES solver, where varying levels of overlaps were used. The GasLib-4197 dataset is used for this analysis, as for the other large-scale GasLib datasets there are certain combinations of number of subdomains and overlap levels where the construction of the AS preconditioner fails, see Section 9.3.2.

The GasLib-4197 dataset is considered large-scale in this research. Therefore, the difference between considering an x-level overlap versus an $(x+1)$-level overlap will be much less prominent than for the small-scale datasets. Additionally, the range of overlap levels possible before at least one subdomain contains the entire network is much larger for the large-scale networks, compared to the small-scale networks. A summary of the effect of varying the overlap level on the initial preconditioned system as well as the linear solver performance is given in Table 9.20 and Table 9.21. The same effect as for small-scale gas networks is observed. Lower levels of overlap (2- and 4-level) result in the highest condition numbers of the initial preconditioned Jacobian out of all considered overlap levels. Additionally, these cases have the highest number of NR iterations where 1 or 2 restarts are required, as well as the highest average number of linear iterations per NR iteration. The relative sub-system sizes of the sub-system Jacobians only have a small range from smallest to largest, relatively close to 25%, which is expected when the network graph is partitioned into 4 subdomains. The small range indicates that the number of nodes in the original subdomain (without overlap) dominates the number of nodes added through overlap, which is a difference compared to small-scale GasLib datasets. When the overlap level is increased to 10-, 20- and 50-level, the condition number decreases, indicating a more effective preconditioner. This is also observed in the linear iterations per NR iteration. Both the number of restarts as well as the number of iterations per NR iteration decreases. The relative sub-system sizes and its range also increase. The cases of 10- and 20-level overlaps allow for more communication between the subdomains compared to lower levels, while the relative sub-system sizes are less than a third and half of the original problem, respectively. Although a 50-level overlap results in the smallest condition number of fewest linear iterations on average, the computational cost associated with the preconditioner construction and linear solution process can get close to using a direct solve. Note that even through 10- and 20-level overlaps appear to balance to benefits of increased RAS performance with added computational costs, the overlap is quite high. Usually in domain decomposition (DD) methods, overlaps not higher than 4 are used in practice.

Table 9.20: Spectral properties and linear solver performance for the preconditioned GasLib-4197 network Jacobian evaluated at $\mathbf{x}_0^g$, using a RAS preconditioner, constructed with a ParMETIS partition of 4 subdomains and varying levels of overlap. Bold values in the linear iterations column indicate a restart.

Overlap	Condition number	Largest SV	Smallest SV	Sub-system size
2-level	$2.64 \cdot 10^7$	$3.98 \cdot 10^2$	$1.51 \cdot 10^{-5}$	25.2% - 26.8%
4-level	$1.69 \cdot 10^7$	$4.04 \cdot 10^2$	$2.39 \cdot 10^{-5}$	25.8% - 27.8%
10-level	$5.87 \cdot 10^6$	$3.61 \cdot 10^2$	$6.15 \cdot 10^{-5}$	27.5% - 33.0%
20-level	$1.19 \cdot 10^6$	$2.24 \cdot 10^2$	$1.88 \cdot 10^{-4}$	32.1% - 48.1%
50-level	$3.25 \cdot 10^5$	$2.07 \cdot 10^2$	$6.38 \cdot 10^{-4}$	66.0% - 87.7%

Table 9.21: Summary of the linear convergence behaviour per NR iteration for the GasLib-4197 dataset, using a RAS preconditioner constructed using a partition into 4 subdomains with the ParMETIS algorithm, and varying levels of overlap. In all cases 13 NR iterations are required. 'Count' refers to the number of NR iterations where the specified number of restarts occurs.

Overlap	Total linear iterations	No restart		1 restart		2 restarts	
		Count	Avg. its.	Count	Avg. its.	Count	Avg. its.
2-level	543	5	27.2	4	36.0	4	65.8
4-level	417	8	22.8	2	35.5	3	54.7
10-level	311	9	19.0	3	33.7	1	39.0
20-level	170	11	11.8	1	19.0	1	21.0
50-level	107	12	7.9	1	12.0	0	-

For small-scale electricity networks, the same effect of varying the overlap-level in the AS preconditioner construction was observed as for small-scale gas networks. An increase in overlap level improves the preconditioner effectiveness, but comes with a higher computational cost. To investigate whether the same effects of changing the overlap level hold for large-scale electricity networks, the PandaPower case6495rte and case9241pegase are considered. The RAS preconditioner is applied to the restarted GMRES solver, where varying levels of overlap are used. Similarly to large-scale gas networks, the range of overlap-levels possible is expected to be larger compared to small-scale electricity networks, due to the larger number of nodes. The summarised results for the case6495rte and case9241pegase datasets are given in Table 9.22, where the relative sub-system sizes and linear convergence history is given. For each overlap level, 6 NR iterations are required for both networks. For the case6495rte network, the number of linear iterations per NR iteration as well as the number of restarts decrease when the overlap level is increased. Additionally, increasing the overlap results in larger relative sub-system Jacobians. The rate of change of the relative sub-system sizes as a function of the overlap level is larger than the large-scale gas network GasLib-4197 in Table 9.20. For the case6495rte dataset, a 6-level overlap results in sub-systems that are already about half of the total system size. This number of nodes included in the overlap starts to dominate from a 6-level overlap onward, indicating a connected network. For the results of the case9241pegase network, the growth in sub-system size when increasing the overlap is not as rapid as for the case6495rte network, but the range of relative sub-system sizes is larger for the largest considered overlap. The cases with a higher overlap require less linear iterations per NR iteration. Differently to the case6495rte network, the number of NR iterations where a restart occurs does not (yet) drop in a similar way for the case9241pegase network.

Table 9.22: Summary of the linear convergence behaviour per NR iteration for the case6495rte and case9241pegase datasets, using a RAS preconditioner constructed using a partition into 4 subdomains with the ParMETIS algorithm, and varying levels of overlap. In all cases 6 NR iterations are required for both networks. 'Count' refers to the number of NR iterations where the specified number of restarts occurs.

Network	Overlap	Sub-system size	Total lin. its.	No restart		1 restart	
				Count	Avg. its.	Count	Avg. its.
case6495rte	2-level	25.5% - 27.9%	176	2	28.0	4	30.0
	4-level	32.6% - 35.7%	124	2	20.0	4	21.0
	6-level	46.1% - 52.1%	93	3	15.0	3	16.0
	8-level	63.0% - 72.2%	67	3	11.0	3	11.3
	10-level	78.2% - 88.4%	54	6	9.0	0	-
case9241pegase	2-level	25.1% - 27.0%	247	0	-	6	41.2
	4-level	27.2% - 30.2%	175	0	-	6	29.2
	6-level	31.0% - 35.6%	134	2	21.0	4	23.0
	10-level	44.7% - 50.1%	109	1	16.0	5	18.6
	20-level	66.4% - 83.2%	74	1	11.0	5	12.6

Number of Subdomains

For small-scale gas and electricity networks, increasing the number of subdomains generally increases the condition number of the initial preconditioned Jacobian, as well as the number of linear iterations per NR iteration. A lower number of subdomains thus results in a more effective preconditioner. On the other hand, decreasing the number of subdomains also increases the size of the local subdomain problems to be solved, potentially increasing the computational cost of constructing the preconditioner. To investigate whether the same effects of changing the overlap level hold for large-scale gas and electricity networks, the GasLib-4197 and case9241pegase datasets are considered, respectively. Both are the largest datasets out of all gas and electricity test networks available for this research. The RAS preconditioner is applied, which is constructing using a partition into varying numbers of subdomains with the ParMETIS algorithm, and applying a 10-level overlap for the GasLib-4197 case, and a 4-level overlap for the case9241pegase network (based on the results in the previous section on overlap levels). The number of subdomains the networks are partitioned into is varied between 2 and 10.

A summary of the effect of varying the number of subdomains on the initial preconditioned Jacobian as well as linear solver performance on the GasLib-4197 dataset is given in Table 9.23. The results observed for the GasLib-4197 dataset is similar compared to small-scale gas networks when varying the number of subdomains. The fewest total number of linear iterations is required when only 2 subdomains are used, but for this case the relative sub-system size are largest, both over 50%. When the number of subdomains is increased, the total number of linear iterations is also increased. Note that the total number of restarts during the linear solves remains roughly the same. The effect of the number of subdomains is not as drastic on the number of linear iterations compared to varying the overlap level. For small-scale gas networks, it was observed that when a large number of subdomains is used, the range of relative sub-system sizes can become quite large (see the last row of Table 9.4). For the GasLib-4197 dataset this is not observed, as the range of relative sub-system size is roughly equal for all considered numbers of subdomains.

Table 9.23: Summary of the linear convergence behaviour per NR iteration for the GasLib-4197 dataset, using a RAS preconditioner constructed using a partition into varying numbers of subdomains with the ParMETIS algorithm, and using a 10-level overlap. In all cases 13 NR iterations are required. 'Count' refers to the number of NR iterations where the specified number of restarts occurs.

Subdom.	Sub-system size	Total lin. its.	No restart		1 restart		2 restarts	
			Count	Avg. its.	Count	Avg. its.	Count	Avg. its.
2	52.1% - 56.3%	234	11	9.0	2	12.5	0	-
4	27.5% - 33.0%	311	9	19.0	3	33.7	1	39.0
8	15.6% - 18.8%	313	10	21.7	3	32	0	-
10	12.1% - 16.8%	359	9	25.1	4	33.3	0	-

A summary of the effect of varying the number of subdomains on the initial preconditioned Jacobian as well as linear solver performance on the case9241pegase dataset is given in Table 9.24. The observed results are similar compared to small-scale electricity networks. The least number of linear iterations is required when only 2 subdomains are considered, but this preconditioner is most expensive to construct due to the large relative sub-system sizes. Note that the number of restarts required decreases when more subdomains are considered, despite an increase in the total number of linear iterations. The opposite effect is observed when varying the level of overlap, where the number of restarts increases when the total number of linear iterations increases. For small-scale electricity networks the range of relative sub-system sizes increases when the number of subdomains is increased. For the case9241pegase dataset this is also observed, although the increase in sub-system size is not as extreme compared to the small-scale datasets. Note that for the large-scale gas network dataset, GasLib-4197 the range of sub-system sizes does not increase when the total number of subdomains is increased. This shows that the underlying partition and network configuration also (indirectly) affect the AS preconditioner cost and performance.

Table 9.24: Summary of the linear convergence behaviour per NR iteration for the case9241pegase dataset, using a RAS preconditioner constructed using a partition into varying numbers of subdomains with the ParMETIS algorithm, and using a 4-level overlap. In all cases 6 NR iterations are required. 'Count' refers to the number of NR iterations where the specified number of restarts occurs.

Subdomains	Sub-system size	Total linear iterations	No restart		1 restart	
			Count	Avg. its.	Count	Avg. its.
2	51.8% - 52.4%	153	0	-	6	25.5
4	27.2% - 30.2%	175	0	-	6	29.2
8	18.6% - 24.1%	185	1	29.0	5	31.2
10	9.8% - 16.4%	206	3	33.0	3	35.7

9.3.4. Reusing the Initial AS Preconditioner

The solution process for solving the steady-state load flow equations of energy networks used in this research consists of the non-linear NR method, where each linearised system is solved using AS preconditioned restarted GMRES. The RAS preconditioner is computed during NR iteration k as follows

$$M^{-1}_{\text{RAS},k} = \sum_{i=1}^{N} {R_i^0}^T A_i^{-1} R_i^\delta,$$
$$= \sum_{i=1}^{N} {R_i^0}^T J(\mathbf{x}_k)_i^{-1} R_i^\delta,$$

where the sub-system Jacobian is a part of the Jacobian matrix evaluated at the current NR iterate. The computation of a preconditioner can be expensive, especially when the systems are large, but also when the computation has to be done repeatedly. To decrease the total computational cost of the solution process, a different target matrix can be considered other than the current Jacobian $J(\mathbf{x}_k)$, as is done in [23]. Taking the initial Jacobian as target matrix results in the same preconditioner for each NR iteration, given as

$$M^{-1}_{\text{RAS},k} = \sum_{i=1}^{N} {R_i^0}^T J(\mathbf{x}_0)_i^{-1} R_i^\delta.$$

Reusing the initial AS preconditioner saves computational costs in constructing the preconditioner during each NR iteration. The expensive factorizations to compute the local updates have to be computed once. When the initial preconditioner is reused, its quality for the later iterations will decrease, as it does not use the updated variables. It is beneficial to reuse the initial AS preconditioner when the cost of the extra iterations required due to a lesser quality preconditioner outweighs the repeated computation of the updated preconditioner.

A different argument to reuse the initial AS preconditioner is to overcome issues during the preconditioner construction occurring in later iterations. Section 9.3.2 describes the situation where the construction of the AS preconditioner is terminated during the second or third NR iteration of the GasLib-582 dataset, due to singular sub-system matrices as a result of zero boundary link flow. Additionally, the GasLib-2607 dataset also presents issues when constructing the AS preconditioner during a later NR iteration for certain AS settings. Reusing the initial AS preconditioner could be a better alternative to using a different or no preconditioner during the linear solution steps.

Gas Networks

The effectiveness and performance of reusing the RAS preconditioner is investigated using the GasLib-582 and GasLib-4197 datasets. The RAS preconditioner is constructed by partitioning the network graph into 4 subdomains with the ParMETIS algorithm, and employing a 2-level overlap. With these settings, the construction of the RAS preconditioner for the GasLib-582 network fails during the third NR iteration due to a singular sub-system matrix (see Section 9.3.2). To study the effect of reusing a previous RAS preconditioner when construction fails during a later iteration, this particular case was chosen. Because the RAS construction of the second NR iteration is successful, it is also investigated what the effects are of reusing this RAS preconditioner during the next NR iterations, given by

$$M^{-1}_{\text{RAS},1} = \sum_{i=1}^{N} {R_i^0}^T J(\mathbf{x}_1)_i^{-1} R_i^{\delta},$$

where the sub-system Jacobians are evaluated at $\mathbf{x}_1$, instead of $\mathbf{x}_0$.

The final residuals of the steady-state load flow equations of the GasLib-582 and GasLib-4197 networks when reusing the initial and second RAS preconditioners, $M^{-1}_{\text{RAS},0}$ and $M^{-1}_{\text{RAS},1}$, respectively, are similar compared to using a direct solve, see Table 9.25. For the GasLib-4197 network, the final residual when reconstructing the RAS preconditioner each NR iteration is also included. When it is mentioned that $M^{-1}_{\text{RAS},1}$ is reused, it is implied that $M^{-1}_{\text{RAS},0}$ is applied during the first NR iteration, and $M^{-1}_{\text{RAS},1}$ during all further NR iterations. From Table 9.25, one can conclude that reusing the initial or second preconditioner does not significantly affect the final residual, compared to using a direct solve for this this dataset. Besides the final residual, the NR residual history of all three considered cases is similar, with only very small differences between them.

Table 9.25: The final residual when solving the steady-state load-flow equations of the GasLib-582 and GasLib-4197 networks, when the linearised systems are solved directly and using restarted GMRES, preconditioned with the initial and first RAS preconditioners $M^{-1}_{\text{RAS},0}$ and $M^{-1}_{\text{RAS},1}$, respectively. The RAS preconditioners were constructed using 4 subdomains following from the ParMETIS algorithm, and 2-level overlap.

Network	Linear solver	Preconditioner	NR iterations	Final residual
GasLib-582	Direct (LU)	-	14	$2.428902 \cdot 10^{-7}$
	Restarted GMRES	$M^{-1}_{\text{RAS},0}$	14	$2.430209 \cdot 10^{-7}$
	Restarted GMRES	$M^{-1}_{\text{RAS},1}$	14	$2.428458 \cdot 10^{-7}$
GasLib-4197	Direct (LU)	-	13	$8.509677 \cdot 10^{-7}$
	Restarted GMRES	$M^{-1}_{\text{RAS},0}$	13	$9.788708 \cdot 10^{-7}$
	Restarted GMRES	$M^{-1}_{\text{RAS},1}$	13	$8.393372 \cdot 10^{-7}$
	Restarted GMRES	$M^{-1}_{\text{RAS},k}$	13	$8.509359 \cdot 10^{-7}$

The linear convergence history of the GasLib-582 case is summarised in Table 9.26, comparing the application of the $M^{-1}_{\text{RAS},0}$ and $M^{-1}_{\text{RAS},1}$ preconditioners. In order for restarted GMRES to converge during each NR iteration, the restart value has to be increased to $m \approx 140$ for $M^{-1}_{\text{RAS},0}$. When the default $m = 100$ is used, the relative preconditioned residual of the second NR iteration stalls and the solver stops only when the maximum number of iterations is reached. This indicates that the Krylov subspace cannot be sufficiently spanned within $m = 100$ iterations. Despite the relative preconditioned residual not converging to the preset tolerance with $m = 100$, the NR residuals still show the same non-linear convergence history, terminating at a final residual of $2.428656 \cdot 10^{-7}$, very similar to the final residuals in Table 9.25.

Table 9.26: Summary of the linear convergence history when solving the steady-state load-flow equations of the GasLib-582 network using restarted GMRES as the linear, preconditioned with the initial and first RAS preconditioners $M^{-1}_{\text{RAS},0}$ and $M^{-1}_{\text{RAS},1}$, respectively. The RAS preconditioners were constructed using 4 subdomains following from the ParMETIS algorithm, and 2-level overlap. The dashed line indicates from which NR iteration onwards the RAS preconditioner is reused.

NR iteration	**Preconditioner:** $M^{-1}_{\text{RAS},0}$		**Preconditioner:** $M^{-1}_{\text{RAS},1}$	
	Linear iterations	**Restarts**	**Linear iterations**	**Restarts**
1	12	-	12	-
2	390	3	12	-
3	385	4	33	-
4	164	2	39	-
5	144	2	67	-
6	203	3	59	1
7	112	2	71	-
8	204	3	85	-
9	132	2	109	-
10	169	2	120	-
11	98	-	235	-
12	89	-	162	2
13	203	2	140	1
14	96	-	131	-
Total	2,401	25	1,275	4

Table 9.27 shows the linear convergence history of the GasLib-4197 case, comparing the application of the $M^{-1}_{\text{RAS},0}$ and $M^{-1}_{\text{RAS},1}$ preconditioners to $M^{-1}_{\text{RAS},k}$. Similarly to the GasLib-582 dataset, the restart value m had to be increased to $m \approx 400$ in order to have linear convergence during each NR iteration. When $m = 100$, the linear iterations of the second NR iteration do not converge. Despite that, the non-linear residual does converge in 13 NR iterations, but showing a slightly different convergence history plot and final residual.

Table 9.27: Summary of the linear convergence history when solving the steady-state load-flow equations of the GasLib-4197 network using restarted GMRES as the linear, preconditioned with the initial and first RAS preconditioners $M^{-1}_{\text{RAS},0}$ and $M^{-1}_{\text{RAS},1}$, respectively. The RAS preconditioners were constructed using 4 subdomains following from the ParMETIS algorithm, and 2-level overlap. The dashed line indicates from which NR iteration onwards the RAS preconditioner is reused.

NR	**Preconditioner:** $M^{-1}_{\text{RAS},0}$		**Preconditioner:** $M^{-1}_{\text{RAS},1}$		**Preconditioner:** $M^{-1}_{\text{RAS},k}$	
iteration	**Linear its.**	**Restarts**	**Linear its.**	**Restarts**	**Linear its.**	**Restarts**
1	37	1	37	1	37	1
2	745	1	30	0	30	0
3	124	1	115	2	30	0
4	138	1	119	2	25	0
5	132	1	139	2	26	0
6	133	1	159	2	25	0
7	247	2	139	2	48	1
8	179	1	167	2	70	2
9	334	2	193	2	62	2
10	407	2	330	2	19	1
11	437	1	283	1	60	2
12	454	1	570	2	71	2
13	641	1	696	1	40	1
Total	4,008	16	2,977	21	543	12

In general, reusing a previous RAS preconditioner in later NR iterations increases the number of linear iterations, compared to reconstructing the preconditioner each NR iteration. When $M^{-1}_{\text{RAS},0}$ is reused, the total number of linear iterations required during the non-linear solution process is about twice as many compared to reusing $M^{-1}_{\text{RAS},1}$, see Table 9.26 and Table 9.27. Additionally, the number of restarts that occurred is higher. The increase in linear iterations as a result of reusing the RAS preconditioner

compared to reconstruction is expected. When the preconditioner is reused, it does not use the most recent state of the variables, deviating more from the global Jacobian inverse at that iteration. The preconditioner is thus less effective for the later NR iterations in which it is reused, resulting in more linear iterations being required for the linearised system to converge. The decrease in total linear iterations when using $M^{-1}_{\text{RAS},1}$ compared to $M^{-1}_{\text{RAS},0}$ is also as expected. The $M^{-1}_{\text{RAS},1}$ preconditioner uses the state $\mathbf{x}_1$ to evaluate the sub-system Jacobian matrices, instead of the initial guess $\mathbf{x}_0$. The $\mathbf{x}_1$ iterate is a physically more accurate state of the system. As a result, the $M^{-1}_{\text{RAS},1}$ preconditioner more accurately approaches the inverse of the global Jacobian at later iterations, compared to the $M^{-1}_{\text{RAS},0}$ preconditioner.

Note that in Table 9.26 and Table 9.27, when $M^{-1}_{\text{RAS},0}$ is reused, the number of linear iterations per NR iteration is highest for the second NR iteration, and decreases for later NR iterations in the GasLib-582 application. When $M^{-1}_{\text{RAS},1}$ is reused, the opposite is observed, where the number of linear iterations per NR iteration increases for later NR iterations. When the reusing of $M^{-1}_{\text{RAS},0}$ and $M^{-1}_{\text{RAS},1}$ preconditioners is applied to the other large-scale gas network datasets, this difference is not always observed.

The same observations in terms of linear iteration counts and convergence are done for the other large-scale GasLib networks and AS preconditioner configurations. The NR residuals are mostly similar compared to using a direct solve, when reconstructing the RAS preconditioner per iteration or reusing $M^{-1}_{\text{RAS},0}$ or $M^{-1}_{\text{RAS},1}$. The effect the preconditioner has is primarily on the linear convergence of restarted GMRES. When a previous preconditioner is reused, the number of linear iterations per NR iteration increases. In some cases, especially observed when $M^{-1}_{\text{RAS},0}$ is reused, the linear solver has not sufficiently converged before the maximum number of iterations has been reached, using the stopping criteria described in Section 8.3.1. Increasing restart value m sufficiently will result in reaching the tolerance before the maximum number of iterations. It is generally also observed that, when possible, reusing a later $M^{-1}_{\text{RAS},k}$ results in fewer total linear iterations as well as restarts required to converge, see Table 9.28. In Figure 9.26, one can see the total number of linear iterations required in the solution process of the GasLib-4197 dataset, when reusing a previous RAS preconditioner. The x-axis shows the j such that $M^{-1}_{\text{RAS},j}$ is reused during all later NR iterations. Based on this plot, one can see that there is a significant decrease in the total number of linear iterations in reusing $M^{-1}_{\text{RAS},3}$, compared to $M^{-1}_{\text{RAS},j}$ with $j < 3$. Afterwards, the decrease in linear iterations stalls.

Table 9.28: The final residual and linear convergence history when solving the steady-state load-flow equations of the GasLib-4197 network, when the linearised systems are solved directly and using restarted GMRES, preconditioned with the previous RAS preconditioners. The RAS preconditioners were constructed using 4 subdomains following from the ParMETIS algorithm, and 2-level overlap. $M^{-1}_{\text{RAS},k}$ refers to reconstructing the RAS preconditioner during each NR iteration.

Preconditioner	Total linear iterations	Total restarts	Final residual
Direct (LU)	-	-	$8.509677 \cdot 10^{-7}$
$M^{-1}_{\text{RAS},0}$	4,008	16	$9.788708 \cdot 10^{-7}$
$M^{-1}_{\text{RAS},1}$	2,997	20	$8.393372 \cdot 10^{-7}$
$M^{-1}_{\text{RAS},2}$	1,639	3	$8.494410 \cdot 10^{-7}$
$M^{-1}_{\text{RAS},k}$	543	12	$8.509359 \cdot 10^{-7}$

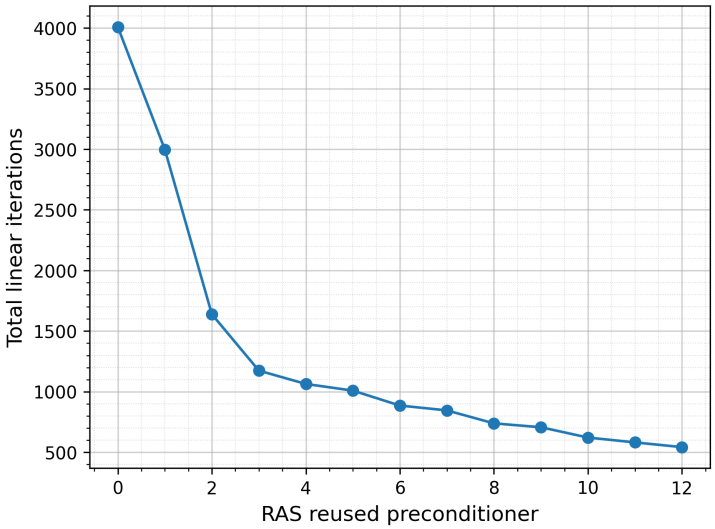


Figure 9.26: The total number of linear iterations in the solution process of the GasLib-4197 dataset using a previous RAS preconditioner, constructed using 4 subdomains with the ParMETIS algorithm and a 2-level overlap

Electricity Networks

For gas networks the motivation of investigating the effectiveness of reusing a previous RAS preconditioner consists of two parts. The first is to prevent termination of the solution process due to a RAS construction failure. Secondly, reusing a previous preconditioner can save computational costs by avoiding repeated expensive factorizations. For the PandaPower electrical networks, a failure of RAS preconditioner construction is not observed. The effectiveness of reusing a previous RAS preconditioner is therefore investigated by comparing its performance to constructing the RAS preconditioner during each NR iteration. This is done by considering the case9241pegase dataset, the largest PandaPower network considered in this research. The RAS preconditioner is constructed using a partition of the network into 4 subdomains with the ParMETIS algorithm, and adding a 2-level overlap. The performance of reusing $M^{-1}_{\text{RAS},0}$ and $M^{-1}_{\text{RAS},1}$ is compared to computing $M^{-1}_{\text{RAS},k}$ for each NR iteration K.

The total number of linear iterations and final residuals of reusing different RAS preconditioners are given in Table 9.29, as well as the final residual when a direct solve in used in each NR iteration. The $M^{-1}_{\text{RAS},k}$ preconditioner indicates that the RAS preconditioner is reconstructed during each NR iteration. There is some variation in the final residual for each reused preconditioner, but they are all still relatively close to the final residual compared to a direct solve. The total number of linear iterations is not significantly impacted when an earlier RAS preconditioner is reused. $M^{-1}_{\text{RAS},2}$ even requires fewer iterations than $M^{-1}_{\text{RAS},k}$, where the RAS preconditioner was reconstructed during each NR iteration.

Table 9.29: The final residual and linear convergence history when solving the steady-state load-flow equations of the case9241pegase network, when the linearised systems are solved directly and using restarted GMRES, preconditioned with the previous RAS preconditioners. The RAS preconditioners were constructed using 4 subdomains following from the ParMETIS algorithm, and 2-level overlap. $M^{-1}_{\text{RAS},k}$ refers to reconstructing the RAS preconditioner during each NR iteration. In all cases, 6 NR iterations were required.

Preconditioner	Total linear iterations	Total restarts	Final residual
Direct (LU)	-	-	$5.769165 \cdot 10^{-11}$
$M^{-1}_{\text{RAS},0}$	254	6	$5.409642 \cdot 10^{-11}$
$M^{-1}_{\text{RAS},1}$	250	6	$5.743893 \cdot 10^{-11}$
$M^{-1}_{\text{RAS},2}$	239	6	$5.511676 \cdot 10^{-11}$
$M^{-1}_{\text{RAS},k}$	247	6	$5.748121 \cdot 10^{-11}$

10

Additive Schwarz Preconditioning for Coupled Multi-Carrier Energy Networks

This chapter presents the results of the load flow analysis when applying the additive Schwarz (AS) preconditioner in the solution process of the steady-state load flow equations of coupled energy networks. The methodology and setup of the performed experiments as well as the implementation of the AS preconditioners for coupled energy networks are discussed in Chapter 8. The results of Chapter 9 show the performance of the AS preconditioner when applied in the solution process of single-carrier (SC) energy networks, both small- and large-scale. In these experiments it was found that the restricted additive Schwarz (RAS) preconditioner variant outperformed the basic and interpolated AS variants. This chapter will therefore focus mainly on the RAS preconditioner.

This chapter starts with Section 10.1, where the general structure and spectrum of the original Jacobian matrices are discussed, as well as convergence. In the rest of the sections, the RAS application to coupled networks is presented. First, the general effects of the AS preconditioner is described in Section 10.2, considering the preconditioned Jacobian structure, spectrum and convergence history. Next, Section 10.3 describes the effect of coupling elements on the conditioning of the preconditioned Jacobian matrices and the solver performance. The overlap is varied in Section 10.4. Finally, Section 10.5 presents the effects network elements of the SC components have on the RAS application to the coupled network.

10.1. Coupled Energy Networks

This section provides general insights into the behaviour and structure of the coupled energy networks considered in this research. The construction of coupled networks from the SC datasets GasLib and PandaPower for gas and electricity, respectively, is given in Section 8.2.2. The aim of this section is to provide insight into the behaviour of a coupled energy networks, focussing on the effects of the coupling units.

10.1.1. Structure of the Coupled Jacobian

The structure of the coupled network consisting of the case-14 and GasLib-11 datasets, where 1 power-to-gas (P2G) unit is used can be found in Figure 10.1. The structure as described in Section 5.1.2 can be seen. Since a P2G coupling unit is used, the electrical node the unit is connected to is changed from a PQ node to a PQV node. As a result, the $\partial \mathbf{F}^e / \partial \mathbf{x}_e$ component is not square. If a gas-fired generator (GFG) were to be used, the gas node the unit is connected to would be changed from a load node to a reference load node, resulting in a non-square $\partial \mathbf{F}^g / \partial \mathbf{x}_g$ component. The effect of the coupling on the global Jacobian matrix structure and the solution process is visualised by reordering the equations and variables.

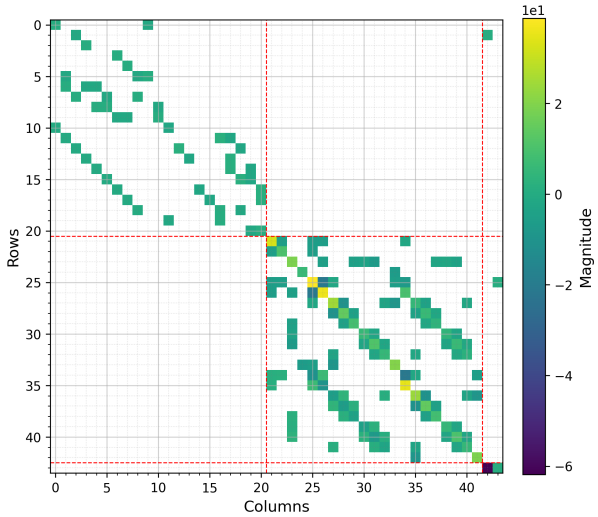


Figure 10.1: Structure heatmap of the global Jacobian of the coupled case14 + GasLib-11 datasets with 1 P2G unit, evaluated at the initial iterate $\mathbf{x}_0$.

Assume that there are n coupling units in total. Denote with n_V the number of coupling units where the connected electrical node is adjusted from a PQ to PQV node by assuming the voltage magnitude $|V|$ known. The number of coupling units where the connected gas node is adjusted from a load node to a reference load node by assuming the nodal pressure p is known, is denoted by n_p. We thus have $n = n_V + n_p$. Similarly, denote the equations related to known voltage magnitude as a result of the coupling with $\mathbf{F}_c^V$, and the equations related to known pressures as $\mathbf{F}_c^p$. The coupling parameters $\mathbf{x}_c$ consist of gas mass flows and real powers of the dummy links. Denote the dummy link variables where the node type change is nodal pressure with $\mathbf{q}_c^p$ and $\mathbf{P}_c^p$. When the node type change is in the voltage magnitude, denote the dummy link variables with $\mathbf{q}_c^V$ and $\mathbf{P}_c^V$. The system of equations associated with the coupled network is reordered such that

$$\mathbf{F}(\mathbf{x}) = \begin{bmatrix} \mathbf{F}_g(\hat{\mathbf{x}}_g, \mathbf{x}_c) \\ \mathbf{F}_c^V(\mathbf{x}_c) \\ \mathbf{F}_e(\hat{\mathbf{x}}_e, \mathbf{x}_c) \\ \mathbf{F}_c^p(\mathbf{x}_c) \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} \hat{\mathbf{x}}_g \\ \mathbf{q}_c^p \\ \mathbf{q}_c^V \\ \hat{\mathbf{x}}_e \\ \mathbf{P}_c^p \\ \mathbf{P}_c^V \end{bmatrix}.$$

The $\hat{\mathbf{x}}_g$ and $\hat{\mathbf{x}}_e$ variables are the gas and electrical variables, respectively, excluding the coupling variables. This reordering leads to the following Jacobian

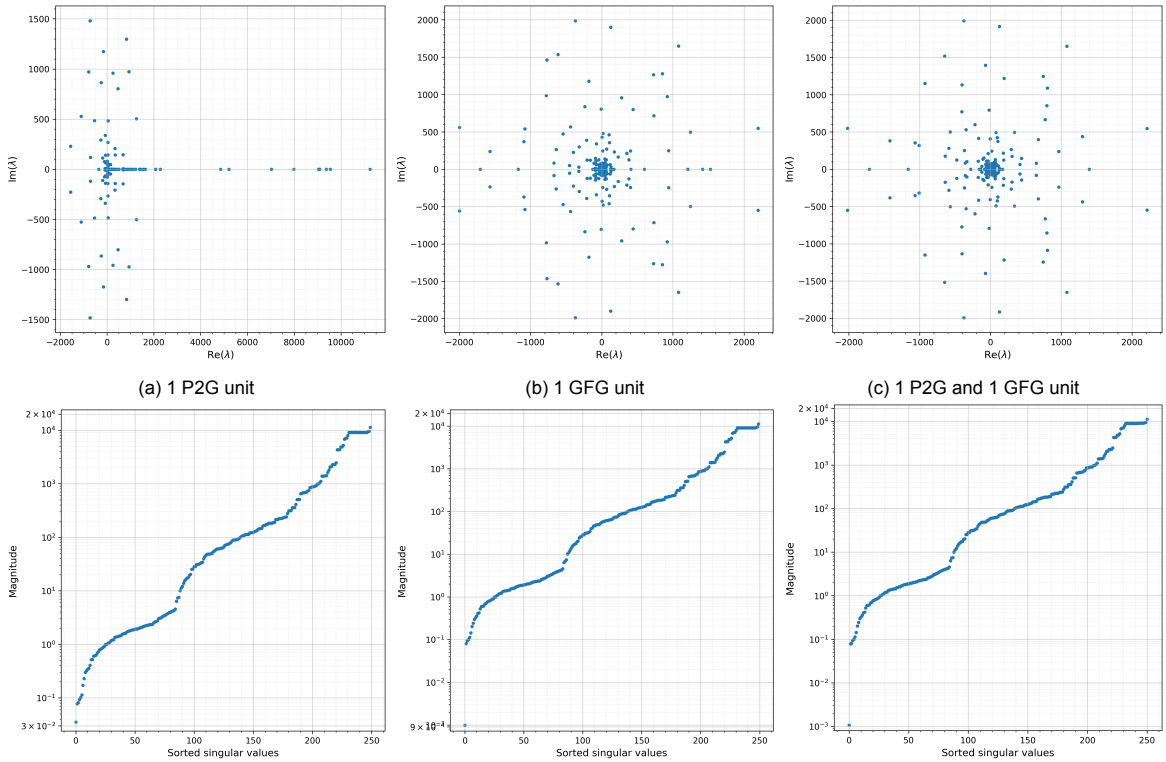
$$J = \begin{bmatrix} \hat{J}_{gg} & \frac{\partial \mathbf{F}_g}{\partial \mathbf{q}_c^p} & \frac{\partial \mathbf{F}_g}{\partial \mathbf{q}_c^V} & 0 & 0 & 0 \\ 0 & 0 & \frac{\partial \mathbf{F}_c^V}{\partial \mathbf{q}_c^V} & 0 & 0 & \frac{\partial \mathbf{F}_c^V}{\partial \mathbf{P}_c^V} \\ 0 & 0 & 0 & \hat{J}_{ee} & \frac{\partial \mathbf{F}_e}{\partial \mathbf{P}_c^p} & \frac{\partial \mathbf{F}_e}{\partial \mathbf{P}_c^V} \\ 0 & \frac{\partial \mathbf{F}_c^p}{\partial \mathbf{q}_c^p} & 0 & 0 & \frac{\partial \mathbf{F}_c^p}{\partial \mathbf{P}_c^p} & 0 \end{bmatrix}. \tag{10.1}$$

The $\hat{J}_{gg}$ and $\hat{J}_{ee}$ components are the non-square Jacobians considering all gas and electricity equations, respectively, excluding the coupling equations and variables. With this reordering, the main diagonal blocks are square. When both $\partial \mathbf{F}_c^V / \partial \mathbf{P}_c^V$ and $\partial \mathbf{F}_c^p / \partial \mathbf{q}_c^p$ are non-zero blocks, there is two-sided interaction within the coupled network. Suppose that $\partial \mathbf{F}_c^V / \partial \mathbf{P}_c^V$ is zero, but $\partial \mathbf{F}_c^p / \partial \mathbf{q}_c^p$ is non-zero, corresponding with one-sided interaction. In Jacobian Equation 10.1, the solution process then consists of solving the upper left block (the gas components), after which the coupling translation is done by lower left component $\partial \mathbf{F}_c^p / \partial \mathbf{q}_c^p$ (the coupling), allowing the lower right block to be solved (the electricity component). Conversely, when only $\partial \mathbf{F}_c^V / \partial \mathbf{P}_c^V$ is non-zero, the solution process consists of solving the

electrical part, then computing the energy flow in the coupling units, after which the gas component is solved.

10.1.2. Spectrum of the Coupled Jacobian

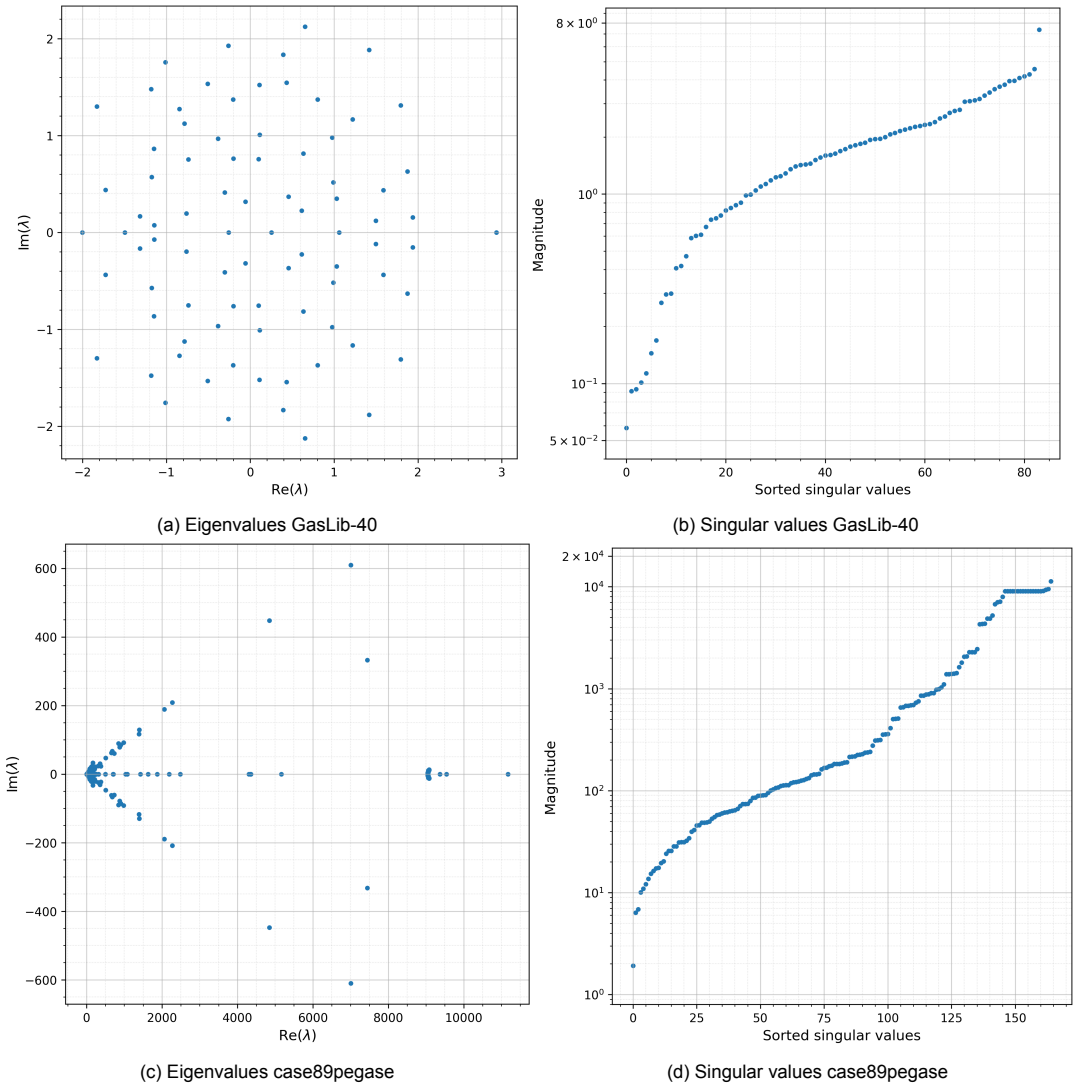
The eigenvalues and singular values of the original Jacobian matrix of the coupled test networks case89pegase and GasLib-40 are given in Figure 10.2, where different coupling configurations are used, showing one-sided and two-sided interaction. The eigenvalues and singular values of the original SC components GasLib-40 and case89pegase are given in Figure 10.3.



(a) 1 P2G unit (b) 1 GFG unit (c) 1 P2G and 1 GFG unit

(d) 1 P2G unit (e) 1 GFG unit (f) 1 P2G and 1 GFG unit

Figure 10.2: Eigenvalues and singular of the global Jacobian of the coupled GasLib-40 and case89pegase networks with different coupling configurations, evaluated at initial iterate $\mathbf{x}_0$.



(a) Eigenvalues GasLib-40

(b) Singular values GasLib-40

(c) Eigenvalues case89pegase

(d) Singular values case89pegase

Figure 10.3: Eigenvalues and singular of the global Jacobians of the SC networks GasLib-40 and case89pegase networks, evaluated at initial iterate $\mathbf{x}_0$.

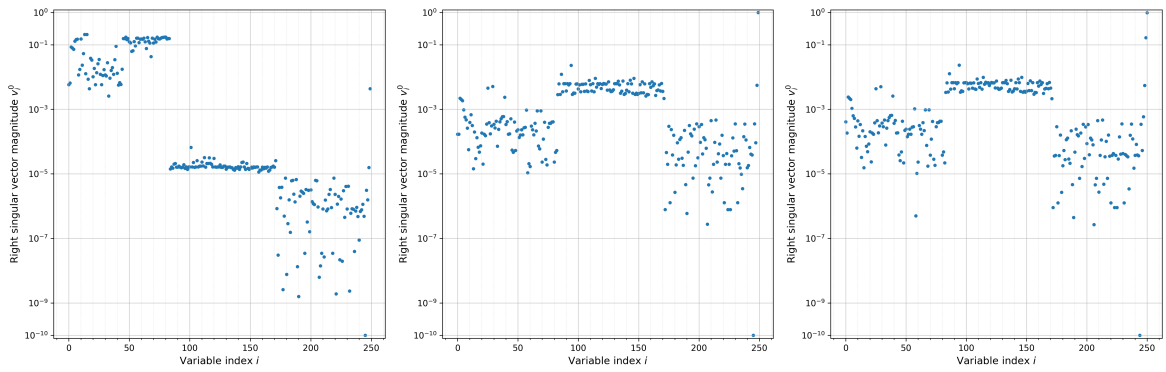
Singular Values

First, consider the singular values of the coupled and SC component networks. The singular values of the GasLib-40 Jacobian range from about $5 \cdot 10^{-2}$ to $8 \cdot 10^{0}$, showing an upward curve, see Figure 10.3b. The singular values of the case89pegase Jacobian range from about $2 \cdot 10^{0}$ to $1 \cdot 10^{4}$, showing a cluster of large singular values with similar magnitude, see Figure 10.3d. The singular values of both these SC components can be recognised in the singular value plots of the coupled networks with different coupling configurations. The differences between the singular values corresponding to the different coupling configurations appear mostly in the smallest singular values. When a GFG is introduced, there is a singular value of about $1 \cdot 10^{-3}$ that is quite smaller than the other smallest singular values. The extreme singular values determine the condition number of the Jacobian matrix. The condition numbers, smallest and largest singular values are given in Table 10.1. The maximum singular value is the same for all coupling configurations, only the minimum singular value is significantly affected. When a GFG unit is included, the condition number increases. This could indicate that coupled networks with GFGs can be more difficult to precondition.

Table 10.1: Spectral properties of the original Jacobian matrix of the coupled GasLib-40 and case89pegase networks, evaluated at $\mathbf{x}_0$, with different coupling configurations.

Coupling	Condition number	Largest SV	Smallest SV
1 P2G	$3.19 \cdot 10^5$	$1.13 \cdot 10^4$	$3.54 \cdot 10^{-2}$
1 GFG	$1.15 \cdot 10^7$	$1.13 \cdot 10^4$	$9.85 \cdot 10^{-4}$
1 P2G and 1 GFG	$1.05 \cdot 10^7$	$1.13 \cdot 10^4$	$1.07 \cdot 10^{-3}$

When there is a singular value that is much smaller than the other (small) singular values, this indicates that perturbations along the direction of the right singular vector corresponding to the smallest singular value barely change the residual $\mathbf{F}(\mathbf{x})$. The right singular vector corresponding to the smallest singular value gives the direction in variable space that results in the smallest change in the system equations. The right singular vector of the smallest singular value of the initial global Jacobian matrix is computed and visualized in Figure 10.4, for all three considered coupling configurations. The x-axis gives the right singular vector index and the y-axis shows the magnitude of all singular vector entries. The magnitude of each entry of the right singular vector indicates the contribution of the corresponding variable to the least sensitive direction in the variable space. When only a P2G unit is coupled, there are multiple variables associated with this direction. When a GFG unit is introduced, there is one variable with a singular vector entry magnitude of almost 1.0. This index is associated with the active coupling power P_c. The large magnitude indicates that the least sensitive direction of the Jacobian is primarily aligned with variations in the active coupling power P_c.



(a) 1 P2G unit (b) 1 GFG unit (c) 1 P2G and 1 GFG unit

Figure 10.4: Right singular vector component magnitudes of the smallest singular value of the global Jacobian of the coupled GasLib-40 and case89pegase networks with different coupling configurations, evaluated at initial iterate $\mathbf{x}_0$.

This occurs because the coupling and reference load node equations introduce a direct relation between gas and electrical system variables, which constrain P_c through multiple equations. As a result, variations in P_c can be partially compensated by corresponding adjustments in the connected variables, leading to a weakly observable direction in the coupled Jacobian. This indicates that the coupling and node change introduce a nearly redundant direction, in which changes in P_c do not produce sufficiently independent residual variations across the full system. Consequently, the system exhibits a near-rank-deficient mode that is associated with the coupling rather than with either SC systems individually.

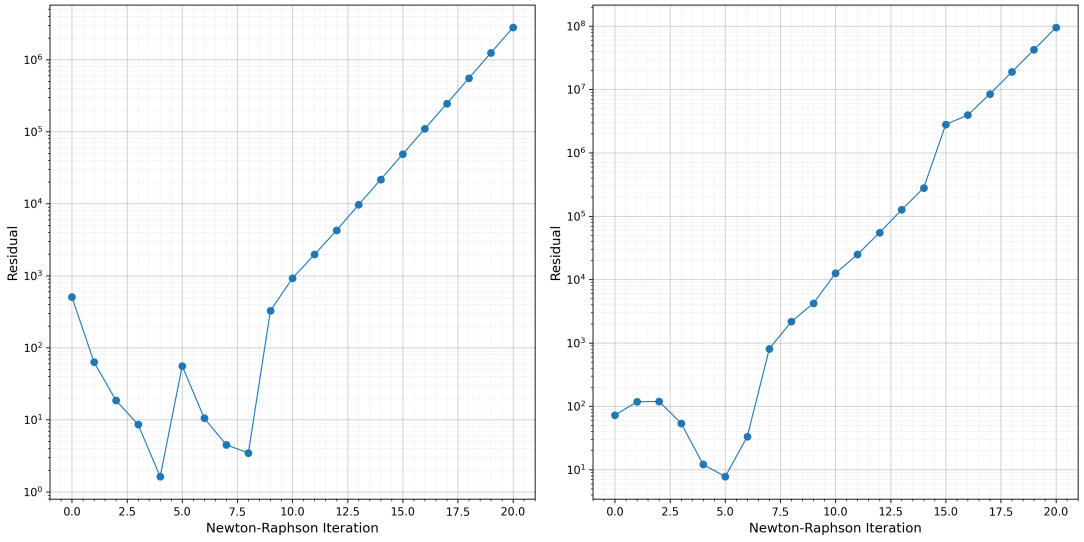
When the node type change is applied in the electrical SC system by fixing the voltage magnitude at the connected electrical node, this near-degenerate behaviour is not observed. In this case, the electrical network retains sufficient internal flexibility through its remaining state variables to absorb perturbations introduced via the coupling. Therefore, variations in the coupling power P_c are distributed across multiple electrical equations rather than collapsing into a single weak direction. This results in a more distributed singular vector structure and a significantly larger smallest singular value, indicating improved observability and reduced redundancy in the coupled system.

Eigenvalues

The eigenvalues of the GasLib-40 and case89pegase networks are spread differently. The GasLib-40 Jacobian eigenvalues are complex and spread in a circle around 0, with a radius of about 2, and one eigenvalue with magnitude 3. The case89pegase Jacobian eigenvalues are complex with real part non-negative, showing a fan shape. There are several smaller clusters of eigenvalues and the largest eigenvalue magnitude is around $1.0 \cdot 10^4$. The magnitudes as well as distribution of the eigenvalues of these networks are different. Unlike the singular values, the spread of the SC eigenvalues is not directly detectable in the eigenvalues of the coupled networks. The eigenvalues of the coupled Jacobian matrix are affected by the coupling configuration. In all three cases, there is a cloud of eigenvalues around zero, with a radius of 1500 to 2000. When only a P2G unit is present, there is a tail of large real-valued eigenvalues, similar to the tail of real-values eigenvalues of the SC case89pegase Jacobian.

10.1.3. Convergence

The coupling of gas and electricity networks is limited by the number of available nodes P2G and GFG units can be connected to, as described in Section 8.2.2. The maximum number of coupling units that can be connected to each GasLib and PandaPower network is given in Appendix C, in Table C.1 and Table C.2 respectively. When a certain coupling configuration is possible between two SC networks, it is not guaranteed that the steady-state load flow equations will converge. In fact, for many coupled networks the residual $\mathbf{F}(\mathbf{x})$ will diverge, see for example Figure 10.5. Both residuals appear to converge during the first iterations, but eventually diverge. It is not within the scope of this research to consider which coupled networks with which coupling configurations converge or diverge and why. In the remainder of this research, the application and performance of the AS preconditioner to coupled networks is only considered for coupled networks that converge.



(a) case89pegase and GasLib-40, with 1 P2G and 2 GFG units

(b) case300 and GasLib-135, with 1 P2G unit

Figure 10.5: Residual behaviour of different coupled gas-electricity networks, using a direct solve for the linearised systems.

When a certain coupling configuration is possible between two networks, and the residual converges when the linearised systems are solved directly, it is possible that the converged dummy link flows are opposite of the defined direction. For example, when a gas and electricity network are coupled using a P2G, the flow of energy should be from the electrical network to the gas network. It is possible that the steady-state load flow converged such that the energy in the P2G unit flows from the gas to the electrical components, which is not possible in practice. During the solution process of coupled networks, it is not enforced that the coupling gas mass flow and real powers must be non-negative. This issue can be overcome by changing the coupling unit type when its flow of energy is opposite of the defined direction, but maintaining the node type change.

10.2. General Effect of AS Preconditioning

This section presents the general effect of the AS preconditioner when applied in the solution process of the steady-state load flow equations of coupled energy networks. The general structure, spectrum and convergence of the coupled networks is described in Section 10.1. The construction of the AS preconditioner for coupled networks is given in Section 8.2.

10.2.1. Structure of the Jacobian

The structure of the RAS preconditioned Jacobian of the coupled network consisting of the case-14 and GasLib-11 datasets, where 1 P2G unit is used, can be found in Figure 10.6. Comparing the structure of the preconditioned Jacobian with the original Jacobian in Figure 10.1, one can see that the preconditioned Jacobian matrix shows a more diagonal structure. All elements on the diagonal are non-zero. The other non-zero elements occur in the $\partial \mathbf{F}^g / \partial \mathbf{x}_e$, $\partial \mathbf{F}^e / \partial \mathbf{x}_e$ and $\partial \mathbf{F}^c / \partial \mathbf{x}_e$ Jacobian components. The columns that contain the non-zero elements correspond to the nodal variable indices of the electricity nodes included in the 1-level overlap.

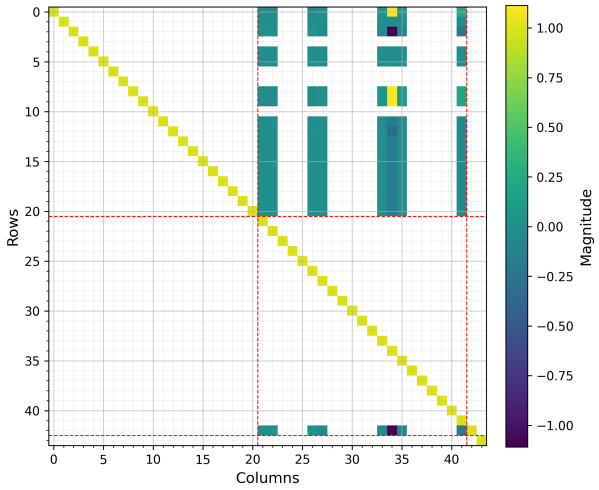
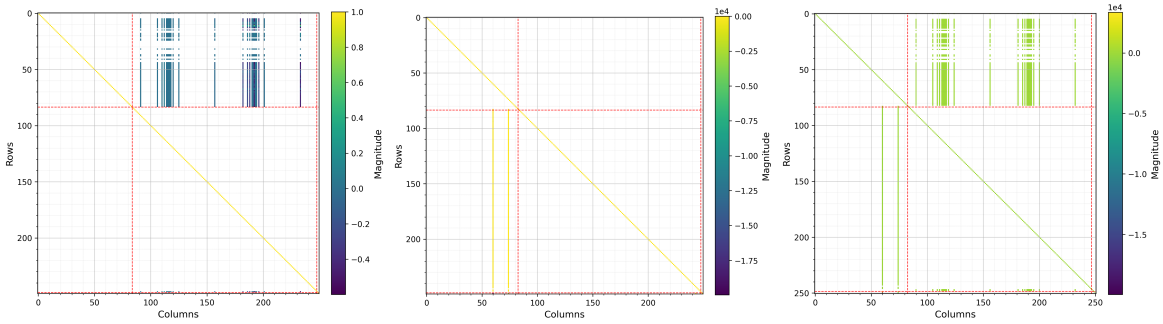


Figure 10.6: Structure heatmaps of the global Jacobian with the RAS preconditioner with 1-level overlap, of the case14 + GasLib-11 network, coupled with one P2G unit, evaluated at initial iterate $\mathbf{x}_0$.

The coupling configuration influences the non-zero elements besides of the diagonal elements in the AS preconditioned global Jacobian, as shown in Figure 10.7. When a P2G unit is coupled, the non-zero elements appear in the columns corresponding to the electrical variable indices, similar to Figure 10.6. When the coupling is done by a GFG unit, the non-zero elements appear in the columns associated with the gas link flow variables. When both coupling unit types are present, the non-zero elements appear in both columns associated with electrical and gas link flow variables.



(a) 1 P2G unit (b) 1 GFG unit (c) 1 P2G and 1 GFG unit

Figure 10.7: Structure heatmaps of the global Jacobian with the RAS preconditioner with 1-level overlap, of the coupled GasLib-40 and case89pegase networks with different coupling configurations, evaluated at initial iterate $\mathbf{x}_0$.

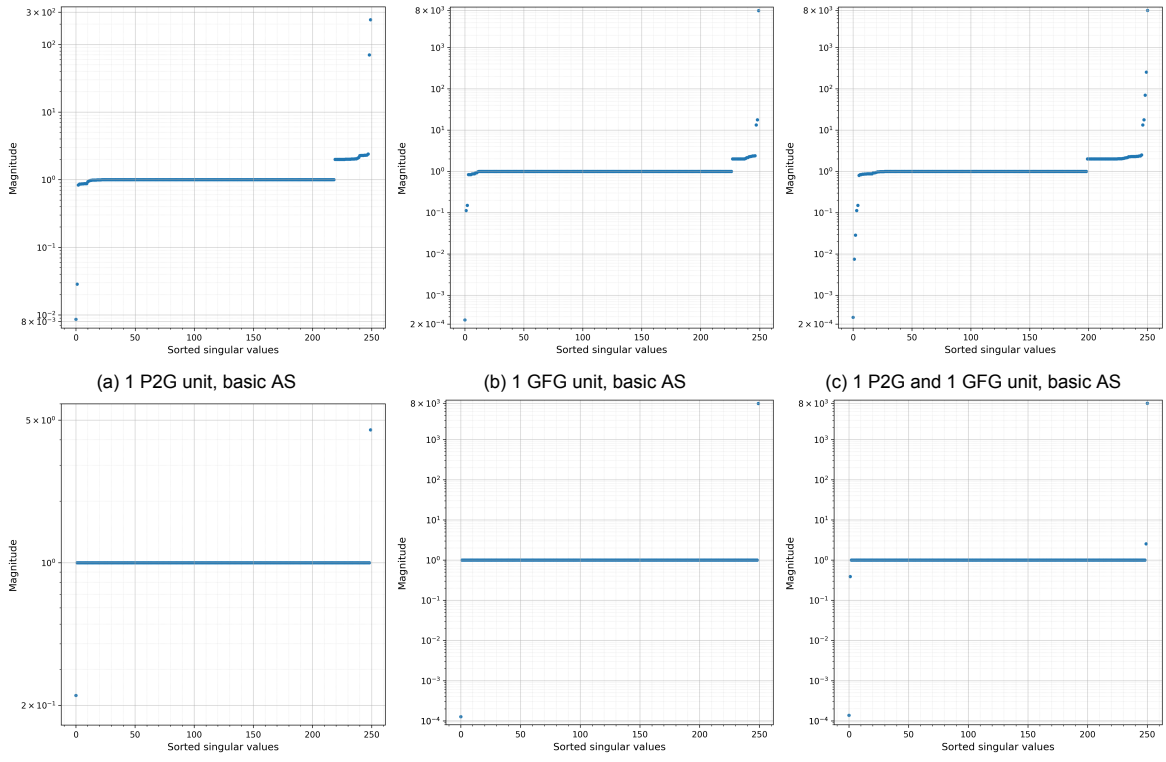
10.2.2. Spectrum of the Jacobian

Figure 10.8 shows the singular values of the basic AS and RAS preconditioned global initial Jacobians of the coupled GasLib-40 and case89pegase network, with different coupling configurations. For both AS types and all three coupling configurations, there are large clusters of singular values exactly equal to 1.

When the basic AS preconditioner is used, there is also a small cluster of singular values exactly equal to 2, for all three coupling configurations. Similarly to the singular values of the basic AS preconditioned SC networks, these singular values of exactly 2 follow from network components being included in both subdomains. When both P2G and GFG units are included, the number of elements included in both subdomains increases, resulting in a larger cluster of singular values of exactly 2. For all coupling configurations, the smallest and largest singular values are more extreme than the middle singular values. When the coupling consists of one P2G unit, there are 4 outliers, 2 small and 2 large singular values. When there is one GFG coupling unit, there are 6 outliers, 3 small and 3 large singular values. When there is a P2G as well as GFG unit present, there are a total of 10 outliers, 5 smaller and 5 larger singular values. These appear to be the smallest and largest singular values associated with either one P2G or GFG unit present. These singular values do not have exactly the same values, but lie very close.

Using the RAS preconditioner, in the case of one coupling unit, there are two singular values not exactly 1, the smallest and the largest, although the value differs depending on the coupling unit. Similarly to the basic AS case, the GFG coupling unit results in extreme singular values of larger magnitude. When both coupling units are included, there are exactly four singular values not exactly equal to 1. They are of similar value as the smallest and largest singular values of the single coupling unit cases.

The largest and smallest singular values, that make up the condition number, are determined by the coupling configuration. A GFG dominates the extreme singular values of the basic AS as well as the RAS preconditioned initial Jacobian. The effect of coupling on the RAS preconditioner is described in more detail in Section 10.3.



(a) 1 P2G unit, basic AS (b) 1 GFG unit, basic AS (c) 1 P2G and 1 GFG unit, basic AS

(d) 1 P2G unit, restricted AS (e) 1 GFG unit, restricted AS (f) 1 P2G and 1 GFG unit, restricted AS

Figure 10.8: Singular values of the basic and RAS preconditioned Jacobians of the coupled GasLib-40 and case89pegase networks with different coupling configurations, evaluated at initial iterate $\mathbf{x}_0$, using a 2-level overlap.

10.2.3. Convergence History

For most converging coupled network datasets considered in this research, the Newton-Raphson (NR) convergence when using the RAS preconditioner is very similar compared to using a direct solve. The linear convergence per NR iteration with the RAS preconditioner is fast, ranging between 1 and 5 linear iterations observed, for both small- and large-scale coupled datasets.

A summary of the linear convergence for the GasLib-40 and case89pegase coupled network is given in Table 10.2, for different coupling configurations. The linear solver is preconditioned with a 2-level overlap RAS preconditioner. The number of linear iterations per NR iteration is very little, ranging from 2 to 4 linear iterations. In the case of 2 GFGs in Table 10.2, where more NR iterations are required compared to other coupling configurations, the number of linear iterations is low. During the last NR iterations of this coupling configuration, only 1 linear iteration is required. This indicates that the RAS preconditioner is effectively the inverse of the Jacobian matrix. The convergence of each linearised system (where at least 2 linear iterations were required) exhibits super-linear convergence.

Table 10.2: Average number of linear iterations per NR iteration of the coupled GasLib-40 and case89pegase networks, with different coupling configurations, using 2-level overlap RAS preconditioned generalized minimum residual (GMRES).

P2G \ GFG	0	1	2
0	1.00	2.00	1.80
1	2.00	3.00	-
2	2.00	4.00	-

Table 10.3 shows the average number of linear iterations required per NR iteration during the solution process of the coupled GasLib-4197 and case9241pegase networks, with various coupling configurations and employing a RAS preconditioner with a 3-level overlap. The total number of NR iterations required varies depending on the coupling configuration, and ranging between 13 and 19. The case where 2 GFG units are considered does not converge. This data shows that the coupling configuration affects the linear convergence history. The introduction of more coupling units increases the average linear iteration count. It appears that the introduction of a GFG unit increases the average linear iteration count more than a P2G. Despite the increase in linear iterations when more coupling units are introduced, an average linear iteration count around 4 is still considered very low for iterative solvers, especially for large-scale systems.

Table 10.3: Average number of linear iterations per NR iteration of the coupled GasLib-4197 and case9241pegase networks, with different coupling configurations, using 3-level overlap RAS preconditioned GMRES.

P2G \ GFG	0	1	2	3	4
0	1.00	2.00	-	2.00	2.00
1	1.47	2.14	2.93	2.36	2.36
2	1.47	3.64	4.14	3.47	4.07
3	1.35	3.13	2.31	4.23	4.21

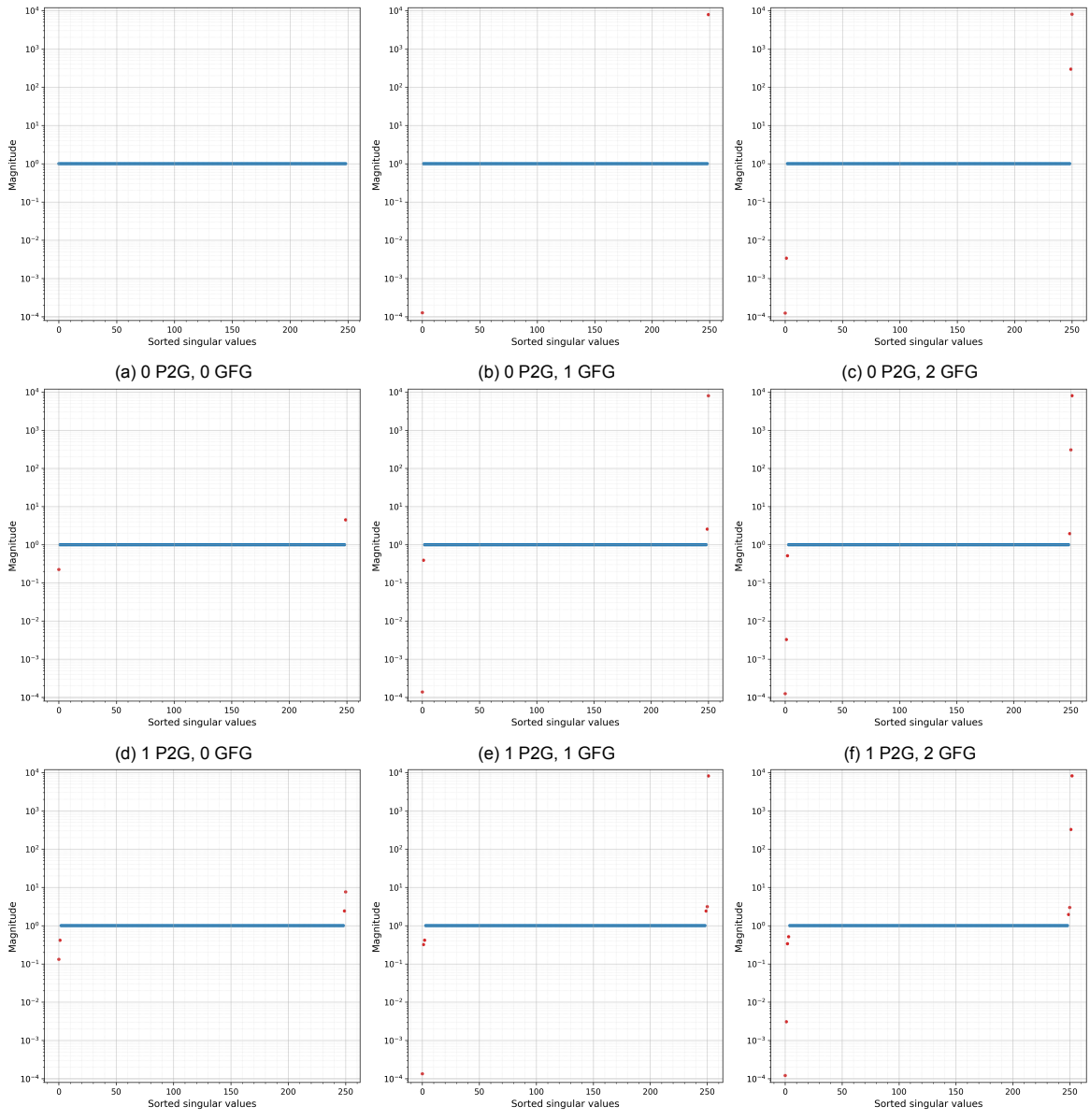
Based the data presented in Table 10.2 and Table 10.3, it appears that a coupled network with one-sided interaction requires less linear iterations for convergence than when there is two-sided interaction. When there is one-sided interaction, the maximum number of average linear iterations per NR iteration for both considered cases is 2. When there is two-sided interaction, the minimum observed average number of linear iterations is 2.14.

10.3. Effect of Coupling

The coupling configuration directly influences the singular values of the preconditioned Jacobian. The effect of the coupling configuration is investigated by considering the RAS preconditioner applied to the (initial) Jacobian of coupled networks, containing n coupling units. The number of P2G units is denoted with n_V, as they require a known voltage magnitude $|V|$. The number of GFG units is denoted with n_p, as they require a fixed nodal pressure p. We thus have $n = n_V + n_p$ as the total number of coupling

units present in the coupled network.

It is observed that the number of singular values σ_i of the RAS preconditioned Jacobian such that $\sigma_i \neq 1$, is always $2n$. More precisely, there are n singular values such that $\sigma_i < 1$, and n singular values such that $\sigma_i > 1$. The order of magnitude of singular value $\sigma_i \neq 1$ is associated with the type of coupling unit. P2G units result in $\sigma_i \neq 1$ that are less extreme than GFG units. The singular values of the RAS preconditioned initial Jacobian of the coupled GasLib-40 and case89pegase network with 9 different coupling configurations are given in Figure 10.9. Per row of images in the figure, the number of P2G units is fixed. Per column, the number of GFG units is kept fixed. Singular values that are not exactly one are indicated with a red colour. Figure 10.9a is the case where no coupling between the networks exist, so that they are solved separately. The RAS preconditioner computes the update of each SC network component directly. This preconditioner is thus exactly the Jacobian inverse.



(a) 0 P2G, 0 GFG (b) 0 P2G, 1 GFG (c) 0 P2G, 2 GFG

(d) 1 P2G, 0 GFG (e) 1 P2G, 1 GFG (f) 1 P2G, 2 GFG

(g) 2 P2G, 0 GFG (h) 2 P2G, 1 GFG (i) 2 P2G, 2 GFG

Figure 10.9: Singular values of the RAS preconditioned Jacobians of the coupled GasLib-40 and case89pegase networks with different coupling configurations, evaluated at initial iterate $\mathbf{x}_0$, using a 2-level overlap. Singular values $\sigma_i = 1$ are given in blue, when $\sigma_i \neq 1$, they are given in red.

Each time a coupling unit is added, a new singular value pair appears, consisting of a σ_i smaller than one, and σ_i larger than one. The singular value pairs associated with GFG units are of larger magnitudes than those associated with P2G units. The largest cluster of singular values larger than one is of size n_p, the second cluster of $\sigma_i > 1$ lying between the largest cluster and 1 is of size n_V. Similarly, the cluster of smallest σ_i is of size n_p, the second cluster of $\sigma_i < 1$ lying between the smallest cluster and 1 is of size n_V.

The exact value of the $\sigma_i \neq 1$ associated with individual coupling units depends on several factors, listed in the paragraphs below. These factors are associated the node type change of the SC node as well as which node the coupling unit is connected to. When the same coupling configuration is used, but the exact value of the voltage magnitude of the PQV electrical node is varied, the associated singular values also vary slightly, but remain of the same order. When the reference pressure of the gas reference load node is adjusted, the associated singular values do not change. The final solution does depend on the reference pressure of the reference load node.

The singular values associated with GFG units are of larger magnitude than those associated with P2G units. Both coupling units are modelled using similar but different equations, namely

$$\mathbf{F}_c^{\text{P2G}} = \eta * P_c - \text{GHV} * q_c,$$
$$\mathbf{F}_c^{\text{GFG}} = P_c - \eta * \text{GHV} * q_c,$$

where GHV is the gross heating value of the gas, and $0 < \eta < 1$ the efficiency of the unit. For all coupling units, the efficiency is set at $\eta = 0.8$. Adjusting the efficiency affects the singular values of the coupling units, but the order of magnitude remains the same.

Additionally, the introduction of other coupling units within the coupled network has an effect on the exact value of the singular values associated with the coupling units. For example, the singular values depicted in Figure 10.9g correspond to the case where 2 P2G units are included. In Figure 10.9h, a GFG is added to the network. A new singular value pair associated with the GFG unit can be observed, but the singular value pairs of the P2G units have also changed, despite there being no change in the coupling with these P2G units.

The difference in order of magnitude of the singular value pairs of P2G units compared to GFG units is not associated with the equations that model these conversion units, but with the node type changes associated with them. When a GFG unit is present in a coupled network, but the node type change is in the voltage magnitude of an electrical node, the magnitude of the singular values is not as extreme and of the same order as the singular value pairs observed with P2G units. Conversely, when a P2G unit is used for coupling, but the pressure of a gas node is assumed known, the singular value pairs are similar to those usually associated with GFG units. Although the coupling unit (equation) influences the value of its corresponding singular values, its order is determined significantly by the node type change.

10.4. Effect of the Overlap

For SC energy networks, the increase of overlap used during the construction of the basic AS or RAS preconditioners improves the effectiveness of the preconditioner. It results in a lower condition number of the initial preconditioned Jacobian and fewer linear iterations required to solve the linearised systems. On the other hand, the cost of construction is more expensive as the local systems are larger. The effect of changing the level of overlap in the RAS application to coupled energy networks is investigated by considering the coupled GasLib-40 and case89pegase networks, with one P2G and one GFG coupling unit.

Table 10.4 shows the condition number and singular values not equal to 1, of the initial RAS preconditioned Jacobian of the coupled GasLib-40 and case89pegase networks, using varying levels of overlap. For each overlap-level, the number of eigenvalues not equal to one remains 4, which is twice the number of coupling units. Increasing the level of overlap results in a reduction of the condition number. Despite the decrease in condition number, the number of linear iterations required to solve

the linearised systems remains the same when the overlap is varied, which is 3 iterations.

Table 10.4: Singular values, condition number and relative sub-system sizes of the coupled GasLib-40 and case89pegase networks, with one P2G and one GFG unit, using RAS preconditioned GMRES with varying overlap levels.

Overlap	Cond. number	Smallest SVs	Largest SVs	Sub-system size
1-level	$3.78 \cdot 10^{8}$	$2.04 \cdot 10^{4}$, $3.27 \cdot 10^{0}$	$3.06 \cdot 10^{-1}$, $5.40 \cdot 10^{-5}$	35.9%, 68.5%
2-level	$5.78 \cdot 10^{7}$	$7.99 \cdot 10^{3}$, $2.56 \cdot 10^{0}$	$3.91 \cdot 10^{-1}$, $1.38 \cdot 10^{-4}$	48.6%, 72.1%
3-level	$6.38 \cdot 10^{7}$	$8.09 \cdot 10^{3}$, $1.62 \cdot 10^{0}$	$6.18 \cdot 10^{-1}$, $1.27 \cdot 10^{-4}$	72.9%, 76.1%
4-level	$2.09 \cdot 10^{7}$	$4.60 \cdot 10^{3}$, $1.19 \cdot 10^{0}$	$8.41 \cdot 10^{-1}$, $2.20 \cdot 10^{-4}$	86.6%, 79.7%

The effect of adjusting the overlap level in the RAS construction is also investigated by applying the RAS preconditioner in the solution process of the GasLib-2607 and case2869pegase network, coupled with one P2G and one GFG unit. Interestingly, the overlap level has an effect on the number of NR iterations required, and therefore the total number of linear iterations. The NR convergence history in the cases of 1-level and 2-level overlap are given in Figure 10.10. The convergence history of the 3-level overlap case is the same as the 2-level overlap case. This difference in NR convergence as a result of the overlap has not been observed for SC energy networks. Up until the 11th NR iteration, the residual is similar for both levels of overlap. After the 11th iteration, the residual of the 1-level case shows super-linear convergence. The 2-level overlap case appears to converge super-linearly until iteration 12, but converges linearly after iteration 12.

Table 10.5: Convergence history of the coupled GasLib-2607 and case2869pegase network, with one P2G and one GFG unit, using RAS preconditioned GMRES with varying overlap levels.

Overlap	NR iterations	Total linear iterations	Final residual
1-level	15	55	$3.620585 \cdot 10^{-7}$
2-level	18	66	$9.950998 \cdot 10^{-7}$
3-level	18	69	$9.774059 \cdot 10^{-7}$

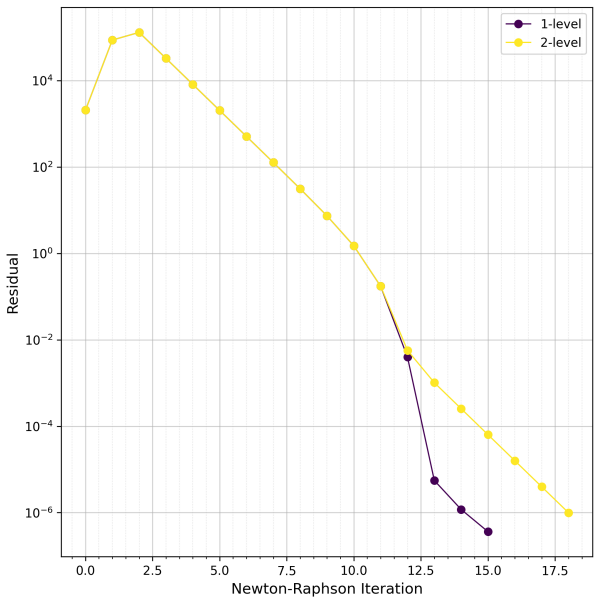


Figure 10.10: NR convergence history of the coupled GasLib-2607 and case2869pegase network with one P2G and one GFG unit. The linearised systems are solved with RAS preconditioned restarted GMRES, using 1- and 2-level overlap.

10.5. Effect of Network Elements

In the GasLib gas network datasets, network components can have several effects on the construction and effectiveness of the AS and RAS preconditioners, see Section 9.1.2 and Section 9.3.2. These

effects also hold for coupled networks in general. In this research the coupled network is always partitioned into two subdomains such that the first subdomain contains all gas network elements, and the second all electrical network elements. As a result, only the second subdomain with all electrical components will have an overlap such that gas network links become boundary links. The construction of the AS and RAS preconditioners for coupled networks can fail as a result of a singular sub-system Jacobian, due to gas network elements on the boundary of the subdomain. The causes for such singular sub-system Jacobians observed in SC gas networks are:

- Compressors: Two or more compressors as boundary links connected to the same node.
- Zero link flow: Two or more links as boundary links connected to the same node that have very small or zero gas mass flow through them.

For coupled networks, there is a new issue that can arise during the AS or RAS preconditioner construction that results in a singular Jacobian matrix. When a GFG is present in the coupled network, the gas node the GFG is connected to is changed from a load node to a reference load node by fixing its nodal pressure. When the reference load node is connected to a compressor or another link element with very small or zero gas mass flow through the link, a singular sub-system Jacobian can occur when a 1-level overlap is used. A compressor is modelled using the following relation between the nodal pressures $p_i = rp_j$, where constant r depends on the compressor unit. There is no gas mass flow variable q_k present in the link equation. In the Jacobian matrix we thus have $\partial \mathbf{F}_k^L / \partial q_k = 0$. The only non-zero components in the row of the Jacobian corresponding to the link flow $\mathbf{F}_k^L$ are $\partial \mathbf{F}_k^L / \partial p_i$ and $\partial \mathbf{F}_k^L / \partial p_j$. When the node type of one connected node, say node i, is changed to a reference load node, the nodal pressure is fixed. Therefore, $\partial \mathbf{F}_k^L / \partial p_i$ no longer appears in the Jacobian matrix. Additionally, when the other connected node j is not contained in the subdomain, its nodal pressure p_j is assumed fixed during the local solve. Therefore, $\partial \mathbf{F}_k^L / \partial p_j$ no longer appears in the Jacobian matrix. As a result, the row in the Jacobian corresponding to link flow $\mathbf{F}_k^L$ is a zero row. A local update can only be computed when the nodal pressures p_i and p_j are such that the compressor equation $p_i = rp_j$ is already satisfied. If not, the local update cannot be computed. To prevent the occurrence of this situation, an overlap level of 2 or higher can be applied. When the overlap is 2 or higher, it cannot occur that a link is a boundary link that is connected to a reference load node.

11

Conclusion and Discussion

This research examines the performance of the additive Schwarz (AS) preconditioner when applied in the solution process of steady-state load flow equations of gas, electricity and coupled energy networks. This methodology and setup of the research is described in Chapter 8. The experiments performed for single-carrier (SC) gas and electricity networks are presented in Chapter 9, where the results are elaborately discussed for small-scale as well as large-scale datasets. The application of the AS preconditioner to coupled networks is analysed in Chapter 10.

This chapter states the conclusions drawn from the experiments and observations done in this research. The general conclusions are given in Section 11.1. The research questions stated in Section 1.3 are answered using the conclusions in Section 11.2. Finally, the limitations and possibilities for further research are discussed and elaborated on in Section 11.3.

11.1. Conclusions

The results and observation of the application of the AS preconditioner to energy networks are presented in Chapter 9 and Chapter 10. From these observations, several conclusions can be drawn that aid in answering the research questions, focussing on the partitioning strategy, the various AS types and coupling of energy networks. Additionally, during the research several observations lead to insightful conclusions on the general application of the AS preconditioner, such as the effect of certain gas network elements and the effect of reusing a previous restricted additive Schwarz (RAS) preconditioner. This section briefly presents the main conclusions from this research.

11.1.1. Gas Network Elements

Gas network link elements can negatively affect the performance of AS preconditioning, by resulting in a preconditioner construction failure and by creating significant errors.

- Resistors are sensitive to pressure differences when computing updates. When acting as a boundary link, the risk of large pressure differences and gas mass flow beyond standard values increases, potentially worsening the preconditioned Jacobian conditioning.
- If two or more compressors are boundary links connected to the same node, the sub-system Jacobian matrix becomes singular, preventing the preconditioner construction.
- If two or more boundary links connected to the same node converge to zero-valued gas mass flow, the sub-system Jacobian matrix becomes singular, preventing the preconditioner construction.
- If a compressor is a boundary link such that the internal node is a reference load node, the preconditioned sub-system Jacobian matrix becomes singular, preventing the preconditioner construction.

It is shown that these issues can be mitigated by choosing a partition such that none of the situations mentioned above can occur. Alternatively, when the preconditioner construction fails during a later

iteration, the most recent constructed preconditioner can be reused in later Newton-Raphson (NR) iterations. When boundary link elements result in large errors, the RAS variant minimises the effect on the preconditioner quality.

11.1.2. Partitioning Strategy

The employed partitioning strategy affects the convergence of the linearised systems solved using AS/RAS preconditioned restarted generalized minimum residual (GMRES) method.

Increasing the number of subdomains increases the number of linear iterations, due to a slower information propagation. With a higher number of subdomains, the number of local updates to compute is also higher, but each local update is cheaper to compute, compared to having fewer subdomains. Increasing the level of overlap decreases the number of linear iterations, due to a faster information propagation, but increases the computation cost of each local update. To achieve accurate and fast load flow solutions, one should thus balance by choosing the minimum number of subdomains, combined with the largest overlap level such that the cost of computation is reasonable.

The partitioning algorithm used to determine the subdomains affects the AS preconditioner performance. A multi-level partitioning algorithm improves the performance of the AS preconditioner compared to an unstructured method. When the proximity of nodes is considered during partitioning, the ratio of internal and overlap nodes is more controllable. Performance differences between ParMETIS and CHACO are network-dependent and negligible.

11.1.3. AS Types

RAS is the preferred variant for energy networks simulations, out of the basic, interpolated and restricted variants, as it generally yields in the lowest number of linear iterations for restarted GMRES to converge. Additionally, boundary elements in gas networks can cause errors that significantly impact the quality of the basic AS and interpolated variants. When RAS is used, these errors are not sufficiently propagated to affect the RAS preconditioner quality.

11.1.4. Reusing a Previous RAS Preconditioner

Reusing a previous RAS preconditioner in a later NR iteration can successfully be applied in the solution process the steady-state load flow equations of SC gas and electricity networks, particularly when RAS preconditioner construction is expensive or fails during a later NR iteration.

- In the application of gas networks, reusing a previous RAS preconditioner increases the number of linear iterations required significantly. Reusing a later RAS preconditioner lowers the total number of linear iterations and restarts. When the second or a later RAS preconditioner is reused, the effect on the final residual is minimal.
- In the application of electricity networks, reusing a previous RAS preconditioner does not increase the number of linear iterations significantly. Additionally, the effect on the final residual is minimal. There are no significant differences on solver performance when reusing the initial or a later RAS preconditioner. It is therefore recommended to reuse the initial RAS preconditioner to reduce the total computational costs.

11.1.5. Coupling

In coupled energy networks, the conditioning of the original and RAS preconditioned Jacobian matrices is worsened when a reference load node is added to the network. The RAS preconditioned initial Jacobian matrix of a coupled network containing n coupling units has n singular values $\sigma_i < 1$ and n singular values $\sigma_i > n$. All other singular values are exactly 1. When more coupling units are introduced in a coupled network, the average number of linear iterations of RAS preconditioned GMRES increases per NR iteration. In a coupled network with two-sided interaction, the average number of linear iterations required for convergence with RAS preconditioned restarted GMRES per NR iteration is higher compared to coupled networks with one-sided interaction.

11.2. Research Questions

This section answers the research questions posed in Section 1.3 based on the results in Chapter 9 and Chapter 10, and conclusions of Section 11.1. First, the four sub-questions are answered, after which the main research question is discussed.

11.2.1. Sub-Questions

The first sub-question is focussed on SC energy networks, and is formulated as:

How does an additive Schwarz preconditioner perform when solving steady-state load flow equations of gas and electricity networks separately?

From the research done in this thesis it was found that the AS preconditioner successfully accelerates the linear convergence of the linearised systems of the gas and electricity networks, compared to not using a preconditioner. The final NR residuals do not significantly differ compared to using a direct solve. The AS preconditioner is algorithmically scalable, where the number of linear iterations grows at a much lower rate than the problem size, maintaining feasibility for large gas and electricity load flow problems. When applying AS to gas networks, one should consider the effects certain gas network elements have on the AS preconditioner quality. For both gas and electricity networks, tuning the AS hyperparameters affects its exact effectiveness.

The second sub-question considers the partitioning strategies, and is given by:

How do different partitioning strategies affect the performance of the additive Schwarz preconditioner when solving steady-state load flow equations of gas, electricity and coupled networks?

Using a structured partitioning algorithm such as ParMETIS and CHACO for the gas and electricity networks improves the AS performance compared to not considering the energy network graph. The differences in AS performance between using the ParMETIS and CHACO algorithms are not significant. The partitioning strategy used for coupled networks in this research is limited to a carrier-based partition. Therefore, the effect of various partitioning strategies for coupled networks cannot yet be quantified.

The third sub-question concerns the AS variants, and states:

How do the basic, interpolated and restricted variants of the additive Schwarz preconditioner compare when solving steady-state load flow equations of gas, electricity and coupled networks?

For gas, electricity and coupled networks, the RAS preconditioner is preferred over the basic AS and interpolated AS variants in terms of performance. The RAS preconditioner generally results in less linear iterations. Additionally, for gas and coupled networks, the RAS variant prevents errors associated with certain boundary link elements to negatively affect the preconditioner quality. In coupled networks, the non-extreme singular values of the RAS preconditioned Jacobian matrix are more tightly clustered around one, compared to using the basic AS preconditioner.

The fourth sub-question focusses on coupled networks, and is stated as:

What is the effect of coupling gas and electricity networks on the performance of the additive Schwarz preconditioner when solving steady-state load flow equations of coupled networks?

The introduction of reference load nodes in the gas network component of a coupled network worsens the conditioning of the original and preconditioned initial Jacobian matrices. A higher number of coupling units generally increases the number linear iterations to solve the linearised system with the RAS preconditioned restarted GMRES solver.

11.2.2. Main Research Question

Recall the main research question:

How does an additive Schwarz preconditioner perform when applied to a Krylov solver in the solution process of the steady-state load flow equations of large scale coupled gas-electricity energy networks?

From the research done in this thesis it was found that the AS preconditioner performs well when applied to restarted GMRES as the linear solver during the solution process of the steady-state load flow equations of large-scale SC and coupled gas-electricity energy networks. The AS preconditioner results in an algorithmically stable solution process, that maintains feasibility for large energy networks. The hyperparameters affect the exact AS preconditioner performance, but most parameter combinations successfully accelerate the solution process compared to an un-preconditioned linear solver. This preconditioning method is robust and not sensitive to parameters choices. The application of AS to coupled networks show good initial results in terms of accuracy, cost and scalability.

11.3. Limitations and Further Research

In addition to the results and conclusions, there are some limitations of this research, and recommendations for further research into AS preconditioning in energy networks, discussed in this section.

11.3.1. Parallel Implementation

The AS preconditioners considered in this research were not implemented using parallelisation techniques. An advantage of using the AS preconditioner is the possibility to execute computations in parallel, improving the speed of the solution process. Although it was not within the scope of this thesis, in further research the current implementation of the AS preconditioner can be adjusted to allow parallel computations in order to investigate its parallel performance. It is expected however, that for the network sizes considered in this research, the parallel implementation will not yet significantly reduce the total solution time. Larger test network datasets are required in order to gain advantage from using parallel computations.

11.3.2. Test Network Data

The AS preconditioner performance was analysed using the GasLib and PandaPower datasets, consisting of synthetic and real-life scenarios and data. These datasets both consist of various networks the preconditioner performance was tested on, including smaller and larger networks. However, the largest networks considered for gas, electricity and coupled networks are not as large as the large-scale networks encountered in industry.

There were several choices made in constructing the coupled test network datasets that limit the variety of coupled networks used in this research. A coupled network consists of one gas component and one electricity component. This choice was made to simplify implementations and to initiate the research into modelling coupled energy networks with relatively simple cases. Coupled networks that consist of multiple gas and/or electricity components were not considered. Future research could focus on extending the coupled network datasets in order to study the performance of various solvers and preconditioners in their solution processes, including the AS preconditioner as constructed in this research.

11.3.3. Partitioning of Single-Carrier Networks

In this research, graph-based partitioning methods were used, that do not consider the underlying physics of the energy networks or the node type. The employed multi-level partitioning algorithms aim to balance the number of nodes in each subdomain. The local sub-systems that are solved during the AS preconditioner construction consist of the subdomains with added overlap. Even when the number of nodes is spread evenly among the subdomains without overlap, the number of Jacobian components in the corresponding sub-systems can vary significantly, especially for higher overlap levels. A partitioning algorithm can be constructed with the aim of balancing the number of components of the sub-systems corresponding to the subdomains. This would imply considering the number of equations and variables associated with each node, as well as considering the overlap during partitioning. Further

research is required to determine such an algorithm, but also to investigate whether the impact of such a partitioning algorithm is significant in the AS preconditioner performance, especially for large-scale networks.

Additionally, in order to prevent certain instances that result in singular sub-system matrices, a partitioning strategy can be considered that takes the gas network components into account during the partitioning process, with the additional aim to prevent issues observed in this research. For example, a partition can be made such that when two or more compressors are connected to the same node, at most one of those can become a boundary link.

11.3.4. Partitioning of Coupled Networks

For this research the choice was made to partition coupled networks based on carrier, in order to initially investigate the effect of coupling on the AS performance. Since all coupled networks consist of one gas network component and one electricity network component, there are always two subdomains when constructing the AS preconditioners. Research into the performance of the AS preconditioner in the solution process of coupled networks can be expanded further by considering other partitioning strategies. Several possibilities are given by:

- The current carrier-based partition of coupled networks can be expanded by further partitioning the SC gas and electricity components. This will result in four types of subdomains. Subdomains with only gas components, subdomains with gas components, gas dummy links and coupling nodes, subdomains with only electrical components and subdomains with electrical components, electrical dummy links and coupling nodes.
- An alternative partitioning strategy can be applied to coupled networks such that the coupled graph structure is considered, not necessarily the carrier type of the nodes. Note that a limitation of this partitioning strategy is that when a coupling node is included in a subdomain, the node with the corresponding node type change should be included in the same subdomain in order for the system to be square. Moreover, new scenarios can arise where certain gas network boundary links can result in singular sub-system Jacobians. Especially the combination of compressors and reference load nodes should be considered.

11.3.5. Linear Solver and Stopping Criteria

The linear solver used in this research is restarted GMRES, specifically the implementation in the `SciPy` library. The stopping criteria used in this solver affect the number of linear iterations, as well as the final residual. In this research, the effect of different stopping criteria and linear solution methods was not investigated due to a focus on the AS preconditioner performance. Further research can consider various stopping criteria, tolerances and linear solvers to further improve computational cost, speed and accuracy of the solution process, as is done in [22], potentially differentiating between carrier type.

11.3.6. Two-Level Additive Schwarz Preconditioning

This research contained the basic, interpolated and restricted AS variants, that differ in the way the local updates are used in the preconditioner construction. Further research can expand the investigation of the AS performance by considering the 2-level AS preconditioner, given as

$$M_{AS,2L}^{-1} = \sum_{i=1}^{N} {R_i^{\delta}}^T A_i^{-1} R_i^{\delta} + \Phi^T A'^{-1} \Phi,$$

where Φ and A' refer to coarse network Jacobian matrices and restriction operators [9, 58]. When the coarse network graph is introduced, the communication between subdomains is likely improved. It can be investigated whether more subdomains of smaller size, whose local updates are cheaper to compute, can be used, without increasing the number of linear iterations compared to not using the coarse correction. Research into this topics has to include determining a method to find an appropriate coarse representation of an energy network graph and corresponding load flow equations.

11.3.7. Modelling of Energy Networks

In modelling of energy networks and translating the physical systems to load flow equations, assumptions and choices have to be made. These modelling choices are made to simplify the governing equations and corresponding computations. This research considers steady-state load flow, where there is no time-dependence. Although it was not within the scope of this research, future research can apply the AS preconditioner to transient energy network models or energy network models where different assumptions were made.

Bibliography

[1] Shrirang Abhyankar, Barry Smith, and Emil Constantinescu. “Evaluation of overlapping restricted additive Schwarz preconditioning for parallel solution of very large power flow problems”. In: *Proceedings of the 3rd International Workshop on High Performance Computing, Networking and Analytics for the Power Grid*. Nov. 2013, pp. 1–8. ISBN: 9781450325103. DOI: 10.1145/2536780.2536784.

[2] Shrirang Abhyankar et al. “Parallel Dynamics Simulation Using a Krylov-Schwarz Linear Solution Scheme”. In: *IEEE Transactions on Smart Grid* 8.3 (May 2017), pp. 1378–1386. ISSN: 19493053. DOI: 10.1109/TSG.2016.2610863.

[3] Charles K. Alexander and Matthew N. O. Sadiku. *Fundamentals of electric circuits*. 5th ed. New York, New York: McGraw-Hill, 2013. ISBN: 9780073380575.

[4] Satish Balay et al. *PETSc/TAO Users Manual*. Tech. rep. ANL-21/39 - Revision 3.24. Argonne National Laboratory, 2025. DOI: 10.2172/2998643.

[5] Kelvin E. Bassey. “Hybrid Renewable Energy Systems Modeling”. In: *Engineering Science & Technology Journal* 4.6 (Dec. 2023), pp. 571–588. DOI: 10.51594/estj/v4i6.1255.

[6] Arthur R. Bergen and Vijay Vittal. *Power System Analysis*. 2nd ed. Upper Saddle River, USA: Prentice Hall, 2000. ISBN: 0136919901.

[7] Xiao-Chuan Cai and Marcus Sarkis. “A Restricted Additive Schwarz Preconditioner for General Sparse Linear Systems”. In: *SIAM Journal on Scientific Computing* 21.2 (Oct. 1999), pp. 792–797. DOI: 10.1137/S106482759732678X.

[8] Xue Chen, Jianwen Cao, and Xuesong Wu. “Research on Efficient Graph Partitioning Algorithms for Sparse Matrices”. In: *2025 5th International Conference on Artificial Intelligence, Big Data and Algorithms (CAIBDA)*. Beijing, China, June 2025, pp. 571–574. ISBN: 9798331526641. DOI: 10.1109/CAIBDA65784.2025.11183378.

[9] Filipe Cumaru, Alexander Heinlein, and Joachim Schöberl. *Two-level additive Schwarz preconditioners for reduced integration methods*. Tech. rep. Delft Institute of Applied Mathematics, Dec. 2025. DOI: 10.48550/arXiv.2512.01706.

[10] Manolo D’Orto et al. “Comparing Different Approaches for Solving Large Scale Power-Flow Problems with the Newton-Raphson Method”. In: *IEEE Access* 9 (Apr. 2021), pp. 56604–56615. ISSN: 21693536. DOI: 10.1109/ACCESS.2021.3072338.

[11] Lisandro D. Dalcin et al. “Parallel distributed computing using Python”. In: *Advances in Water Resources* 34.9 (Sept. 2011), pp. 1124–1139. ISSN: 03091708. DOI: 10.1016/j.advwatres.2011.04.013.

[12] Francisco De León and Adam Semlyen. “Iterative solvers in the Newton power flow problem: Preconditioners, inexact solutions and partial Jacobian updates”. In: *IEE Proceedings-Generation, Transmission and Distribution* 149.4 (July 2002), pp. 479–484. ISSN: 13502360. DOI: 10.1049/ip-gtd:20020172.

[13] Ron S. Dembot, Stanley C. Eisenstatt, and Trond Steihaug. “Inexact Newton Methods”. In: *SIAM Journal on Numerical Analysis* 19.2 (Apr. 1982), pp. 400–408. DOI: 10.1137/0719025.

[14] Xu Deng and Tao Lv. “Power system planning with increasing variable renewable energy: A review of optimization models”. In: *Journal of Cleaner Production* 246 (Feb. 2020), p. 118962. ISSN: 09596526. DOI: 10.1016/j.jclepro.2019.118962.

[15] Reinhard Diestel. *Graph Theory*. 6th ed. Berlin, Germany: Springer Berlin, 2024. ISBN: 9783662701065. DOI: 10.1007/978-3-662-70107-2.

[16] Georg Feulner. “Global Challenges: Climate Change”. In: *Global Challenges* 1 (Sept. 2015), pp. 5–6. DOI: 10.1002/gch2.1003. URL: http://www.ncdc.noaa.gov/sotc/global/201412..

[17] Mikalai Filonchyk et al. “Greenhouse gases emissions and global climate change: Examining the influence of CO2, CH4, and N2O”. In: *Science of the Total Environment* 935 (July 2024), p. 173359. ISSN: 18791026. DOI: 10.1016/j.scitotenv.2024.173359.

[18] Andreas Frommer and Daniel B. Szyld. “An Algebraic Convergence Theory for Restricted Additive Schwarz Methods Using Weighted Max Norms”. In: *SIAM Journal on Numerical Analysis* 39.2 (Apr. 2001), pp. 463–479. DOI: 10.1137/S0036142900370824.

[19] Philipp Gerstner et al. *A Domain Decomposition Approach for Solving Dynamic Optimal Power Flow Problems in Parallel with Application to the German Transmission Grid*. Tech. rep. Heidelberg, Germany: Engineering Mathematics and Computing Lab, Nov. 2016.

[20] Egor Grishin, Grigorii Gerasimov, and Elena Gryazina. “Overview of Python Power Flow Solvers”. In: *2024 6th International Youth Conference on Radio Electronics, Electrical and Power Engineering (REEPE)*. Moscow, Russia: Institute of Electrical and Electronics Engineers Inc., Feb. 2024. ISBN: 9798350382891. DOI: 10.1109/REEPE60449.2024.10479730.

[21] Bruce Hendrickson and Robert Leland. *The Chaco User’s Guide*. Tech. rep. Alburquerque, USA: Sandia National Laboratories, Nov. 1993.

[22] Reijer Idema. “Newton-Krylov Methods in Power Flow and Contingency Analysis”. PhD thesis. Delft University of Technology, Nov. 2012. ISBN: 9789461915382. DOI: 10.4233/uuid:c9c1c1b8-59f5-4934-88a6-345ebc8ce7f9.

[23] Reijer Idema et al. “Scalable Newton-Krylov solver for very large power flow problems”. In: *IEEE Transactions on Power Systems* 27.1 (Feb. 2012), pp. 390–396. ISSN: 08858950. DOI: 10.1109/TPWRS.2011.2165860.

[24] Reijer Idema et al. “Towards faster solution of large power flow problems”. In: *IEEE Transactions on Power Systems* 28.4 (Oct. 2013), pp. 4918–4925. ISSN: 08858950. DOI: 10.1109/TPWRS.2013.2252631.

[25] IPCC. “Special Report: Global Warming of 1.5 ºC, Summary for Policymakers”. In: *Global Warming of 1.5°C*. Cambridge, UK: Cambridge University Press, 2018, pp. 3–24. DOI: 10.1017/9781009157940.001.

[26] Mengshuo Jia et al. “An Investigation on the Applicability of the Integrated Method for Multi-Carrier Energy Flow Analysis”. In: *2018 IEEE Power & Energy Society General Meeting (PESGM)*. Portland, USA, Aug. 2018, pp. 1–5. ISBN: 9781538677032. DOI: 10.1109/PESGM.2018.8585831.

[27] Mazaher Karimi et al. “Application of Newton-based load flow methods for determining steady-state condition of well and ill-conditioned power systems: A review”. In: *Electrical Power and Energy Systems* 113 (Dec. 2019), pp. 298–309. ISSN: 01420615. DOI: 10.1016/j.ijepes.2019.05.055.

[28] George Karypis and Vipin Kumar. “Parallel multilevel k-way partitioning scheme for irregular graphs”. In: *1996 ACM/IEEE Conference on Supercomputing*. Pittsburgh, USA, Nov. 1996. DOI: 10.1145/369028.369103.

[29] George Karypis and Vipin Kumar. “A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs”. In: *SIAM Journal on Scientific Computing* 20.1 (Aug. 1998), pp. 359–392. DOI: 10.1137/S1064827595287997.

[30] George Karypis and Vipin Kumar. “Multilevel k-way Partitioning Scheme for Irregular Graphs”. In: *Journal of Parallel and Distributed Computing* 48.1 (Oct. 1998), pp. 96–129. DOI: 10.1006/jpdc.1997.1404.

[31] James L. Kirtley. *Electric power principles: sources, conversion, distribution, and use*. John Wiley & Sons, Aug. 2010. ISBN: 9780470686362. DOI: 10.1002/9781119994404.

[32] *Klimaatwet*. Dutch. Tech. rep. Dutch Government, 2019. URL: https://wetten.overheid.nl/BWBR0042394/2023-07-22.

[33] Thorsten Koch et al. *Evaluating Gas Network Capacities*. Philadelphia, USA: Society for Industrial and Applied Mathematics, 2015. ISBN: 9781611973686. DOI: 10.1137/1.9781611973693.

[34] Marieke E. Kootte, Baljinnyam Sereeter, and Cornelis Vuik. “Solving the Steady-State Power Flow Problem on Integrated Transmission-Distribution Networks: A Comparison of Numerical Methods”. In: *2020 IEEE PES Innovative Smart Grid Technologies Europe (ISGT-Europe)*. Virtual, Oct. 2020. ISBN: 9781728171005. DOI: 10.1109/ISGT-Europe47291.2020.9248852.

[35] Anne S. Markensteijn. “Mathematical models for simulation and optimization of multi-carrier energy systems”. PhD thesis. Delft University of Technology, July 2021. DOI: 10.4233/uuid:044c083a-4eb1-4999-b789-db7150c4c7df.

[36] Anne S. Markensteijn, Johan E. Romate, and Cornelis Vuik. “A graph-based model framework for steady-state load flow problems of general multi-carrier energy systems”. In: *Applied Energy* 280 (Dec. 2020), p. 115286. ISSN: 0306-2619. DOI: 10.1016/J.APENERGY.2020.115286.

[37] Xiangyang Men et al. “Coupling analysis of gas□electric hybrid system based on Newton–Raphson method”. In: *The 6th International Conference on Renewable Power Generation (RPG)*. Vol. 2017. 13. Wuhan, China, Oct. 2017, pp. 1505–1510. DOI: 10.1049/joe.2017.0582.

[38] Hiroyuki Mori and Fumitaka Iizuka. “An ILU(p)-Preconditioner Bi-CGStab Method for Power Flow Calculation”. In: *2007 IEEE Lausanne Power Tech*. Lausanne, Switzerland, July 2007, p. 2235. ISBN: 9781424421909. DOI: 10.1109/PCT.2007.4538533.

[39] Buu-Van Nguyen, Johan Romate, and Cornelis Vuik. “Modeling an Electrolyzer in a Graph-Based Framework”. In: *Energies* 18 (Feb. 2025), p. 729. ISSN: 1996-1073. DOI: 10.3390/EN18030729.

[40] Buu-Van Nguyen, Johan Romate, and Cornelis Vuik. “Performance of Krylov solvers for gas networks”. In: *The 16th ACM International Conference on Future and Sustainable Energy Systems (E-ENERGY ’25),* Rotterdam, The Netherlands, June 2025, pp. 489–494. ISBN: 9798400711251. DOI: 10.1145/3679240.3734600.

[41] Andrzej J. Osiadacz. *Simulation and analysis of gas networks*. London, UK: E. & F.N. Spon, 1987. ISBN: 9780419124801.

[42] François Pellegrini. “Distillating knowledge about SCOTCH”. In: *Combinatorial Scientific Computing*. Schloss Dagstuhl, Germany, Feb. 2009, pp. 1–12. DOI: 10.4230/DagSemProc.09061.9.

[43] José E.O. Pessanha et al. “Critical investigation of preconditioned GMRES via incomplete LU factorization applied to power flow simulation”. In: *Electrical Power and Energy Systems* 33.10 (Dec. 2011), pp. 1695–1701. ISSN: 01420615. DOI: 10.1016/j.ijepes.2011.08.010.

[44] Sen Qi, Gengfeng Li, and Zhaohong Bie. “Hybrid Energy Flow Calculation for Electric-Thermal Coupling System Based on Inexact Newton Method”. In: *2019 IEEE Sustainable Power and Energy Conference (iSPEC)*. Beijing, China, Nov. 2019. ISBN: 9781728149301. DOI: 10.1109/iSPEC48194.2019.8975152.

[45] Yue Qiu et al. “Efficient Numerical Methods for Gas Network Modeling and Simulation”. In: *Networks and Heterogeneous Media* 15.4 (Nov. 2020), pp. 653–679. DOI: 10.3934/nhm.2020018.

[46] Hassan Qudrat-Ullah. “Modelling and Simulation in Service of Energy Policy”. In: *The 7th International Conference on Applied Energy – ICAE2015*. Vol. 75. Abu Dhabi, UAE, Mar. 2015, pp. 2819–2825. DOI: 10.1016/j.egypro.2015.07.558.

[47] Lukas Razik et al. “A comparative analysis of LU decomposition methods for power system simulations”. In: *2019 IEEE Milan PowerTech*. Milan, Italy, June 2019, pp. 1–6. DOI: 10.1109/PTC.2019.8810616.

[48] Matheus M. Roque et al. “Numerical Investigation on the Solution of Ill-Conditioned Load Flow Linear Equations”. In: *Journal of Control, Automation and Electrical Systems* 30 (Mar. 2019), pp. 580–588. ISSN: 21953899. DOI: 10.1007/s40313-019-00461-2.

[49] Yousef Saad. *Iterative methods for sparse linear systems*. 2nd ed. Society for Industrial and Applied Mathematics, 2003, p. 528. ISBN: 0898715342.

[50] Yousef Saad and Martin H Schultz. “GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems”. In: *SIAM Journal on Scientific and Statistical Computing* 7.3 (July 1986), pp. 856–869. DOI: 10.1137/0907058.

[51] Pieter Schavemaker and Lou van der Sluis. *Electrical power system essentials*. 2nd ed. Chichester, West Sussex: John Wiley & Sons, 2017. ISBN: 9781118803479.

[52] Martin Scheepers et al. “Towards a climate-neutral energy system in the Netherlands”. In: *Renewable and Sustainable Energy Reviews* 158 (Jan. 2022), p. 112097. ISSN: 18790690. DOI: 10.1016/j.rser.2022.112097.

[53] Martin Schmidt et al. “GasLib - A library of gas network instances”. In: *Data* 2.40 (Dec. 2017). ISSN: 23065729. DOI: 10.3390/data2040040.

[54] Maria Sosonkina et al. “A New Adaptive GMRES Algorithm for Achieving High Accuracy”. In: *Numerical Linear Algebra with Applications* 5 (Dec. 1998), pp. 275–297. DOI: 10.1002/(SICI)1099-1506(199807/08)5:4<275::AID-NLA131>3.0.CO;2-B.

[55] Benjamin K. Sovacool. “The intermittency of wind, solar, and renewable electricity generators: Technical barrier or rhetorical excuse?” In: *Utilities Policy* 17.3-4 (July 2009), pp. 288–296. ISSN: 09571787. DOI: 10.1016/j.jup.2008.07.001.

[56] Shriram Srinivasan and Kaarthik Sundar. “The Solution of Potential-Driven, Steady-State Nonlinear Network Flow Equations via Graph Partitioning”. In: (Nov. 2025). DOI: 10.1109/LCSYS.2026.3674163.

[57] Leon Thurner et al. “Pandapower - An Open-Source Python Tool for Convenient Modeling, Analysis, and Optimization of Electric Power Systems”. In: *IEEE Transactions on Power Systems* 33.6 (Nov. 2018), pp. 6510–6521. ISSN: 08858950. DOI: 10.1109/TPWRS.2018.2829021.

[58] Andrea Tosseli and Olof B. Widlund. *Domain Decomposition Methods - Algorithms and Theory*. Springer Berlin, 2005.

[59] United Nations Framework Convention on Climate Change. *Paris Agreement*. Tech. rep. Paris, France: United Nations Framework Convention on Climate Change (UNFCCC), Dec. 2015. URL: https://treaties.un.org/pages/ViewDetails.aspx?src=TREATY&mtdsg_no=XXVII-7-d&chapter=27&clang=_en.

[60] Henk A. van der Vorst. “Bi-CGSTAB: A Fast and Smoothly Converging Variant of Bi-CG for the Solution of Nonsymmetric Linear Systems”. In: *SIAM Journal on Scientific and Statistical Computing* 13.2 (Mar. 1992), pp. 631–644. DOI: 10.1137/0913035.

[61] Lucy Westerweel. *AS Preconditioning in Energy Networks*. June 2026. URL: https://github.com/LucyWesterweel/AS_Preconditioning_Energy_Networks.

[62] Jianzhang Wu et al. “Energy Flow Calculation Method for Multi-Energy Systems: A Matrix Approach Considering Alternative Gas Injection and Dynamic Flow Direction”. In: *Applied Sciences* 15.9 (Apr. 2025), p. 4815. ISSN: 20763417. DOI: 10.3390/app15094815.

[63] Emrullah F. Yetkin and Oguzhan Ceylan. “Recycling Newton - Krylov algorithm for efficient solution of large scale power systems”. In: *Electrical Power and Energy Systems* 144 (Sept. 2022), p. 108559. ISSN: 01420615. DOI: 10.1016/j.ijepes.2022.108559.

[64] Suhan Zhang et al. “Partitional decoupling method for fast calculation of energy flow in a large-scale heat and electricity integrated energy system”. In: *IEEE Transactions on Sustainable Energy* 12.1 (Jan. 2021), pp. 501–513. ISSN: 19493037. DOI: 10.1109/TSTE.2020.3008189.

[65] Mengting Zhu et al. “An integrated multi-energy flow calculation method for electricity-gas-thermal integrated energy systems”. In: *Protection and Control of Modern Power Systems* 6.5 (Feb. 2021), pp. 1–2. ISSN: 23670983. DOI: 10.1186/s41601-021-00182-2.

A

The Bi-Conjugate Gradient (Bi-CG) and Bi-Conjugate Gradient Stabilized (Bi-CGSTAB) Methods

The Bi-CG and Bi-CGSTAB methods are both Krylov subspace methods (see Section 5.3.1), where the computation of the orthogonal basis of the Krylov subspace is based on Lanczos' bi-orthogonalisation method. This Appendix first provides the Lanczos' bi-orthogonalisation method, after which the Bi-CG and Bi-CGSTAB method are derived.

A.1. Lanczos' Bi-Orthogonalisation Algorithm

Lanczos' bi-orthogonalisation algorithm forms dual bases of the right subspace $\mathcal{K}_k(A, \mathbf{r}_0)$ and left subspace $\mathcal{K}_k(A^T, \mathbf{r}_0)$ that satisfy the bi-orthogonality condition where each left vector $\mathbf{w}_i$ is orthogonal to all right vectors $\mathbf{v}_j$, except $\mathbf{v}_i$. This is done by multiplying previous basis vectors $\mathbf{v}_k$ and $\mathbf{w}_k$ by A and A^T, respectively. As a result of the bi-orthogonality, $A\mathbf{v}_k$ and $A^T\mathbf{w}_k$ is spanned by a linear combination of the previous two basis vectors and a new residual component. These new vectors are computed and orthogonalized against the previous two vectors in each sequence. As a result, the dual bases V_k and W_k satisfy the relation $W_k^T V_k = I$. The bi-Lanczos algorithm results in the following relations,

$$AV_k = V_k T_k + \delta_{k+1}\mathbf{v}_k e_k^T, \quad A^T W_k = W_k T_k^T + \beta_{k+1}\mathbf{w}_k e_k^T, \quad W_k^T A V_k = T_k, \tag{A.1}$$

where T_k is a tri-diagonal matrix. The bases V_k and W_k are dual and as a result, the projection of the coefficient matrix A from the oblique projection process onto $\mathcal{K}_k(A^T, \mathbf{r}_0)$ and orthogonally to $\mathcal{K}_k(A, \mathbf{r}_0)$, is a tri-diagonal matrix. It follows that solving systems with T_k is cheaper than solving systems with A.

A.2. The Bi-Conjugate Gradient (Bi-CG) Method

The *bi-conjugate gradient (Bi-CG)* method is a Krylov subspace method used for solving linear systems of equations. The iterations are based on Lanczos' bi-orthogonalization method. The algorithm therefore not only solves the system $A\mathbf{x} = \mathbf{f}$, but also the dual system $A^T\mathbf{x}^* = \mathbf{f}^*$. It is a projection method, taking $\mathcal{K} = \mathcal{K}_k(A, \mathbf{v}_0)$, the Krylov subspace, and $\mathcal{L} = \mathcal{K}_k(A^T, \mathbf{w}_0)$. The procedure for computing the k-th iterate with Bi-CG is given in Algorithm 7. The algorithm makes use of the bi-orthogonality properties in computing the next iterate. The relation in (5.12) also holds here and therefore the k-th iterate is computed as

$$\mathbf{x}_k = \mathbf{x}_0 + V_k\mathbf{y}_k = \mathbf{x}_0 + V_k T_k^{-1}(\beta\mathbf{e}_1), \tag{A.2}$$

where T_k is tri-diagonal. Denote the LU factorization of T_k as $T_k = L_k U_k$. A substitution of this LU factorization into (A.2) leads to

$$\mathbf{x}_k = \mathbf{x}_0 + P_k\mathbf{z}_k, \tag{A.3}$$

where $P_k = V_k U_k^{-1}$ and $\mathbf{z}_k = L_k^{-1}(\beta \mathbf{e}_1)$. Due to the short recurrences from the bi-orthogonality property, the matrix P_k is structured as

$$P_k = [P_{k-1}, \mathbf{p}_k],$$

where column $\mathbf{p}_k$ can be computed from the previous $\mathbf{p}_{k-1}$ and $\mathbf{v}_k$. Similarly, the vector $\mathbf{z}_k$ is structured as

$$\mathbf{z}_k = \begin{bmatrix} \mathbf{z}_{k-1} \\ \zeta_k \end{bmatrix},$$

where entry ζ_k can be computed from ζ_{k-1}. As a result, (A.3) is equivalent to

$$\mathbf{x}_k = \mathbf{x}_{k-1} + \zeta_k \mathbf{p}_k,$$

so the next iterate $\mathbf{x}_k$ can be computed using the previous iterate.

From the derivations above and the bi-orthogonality we thus have the following original and dual identities

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{x}_k + \alpha_k \mathbf{p}_k, \quad \mathbf{r}_{k+1} = \mathbf{r}_k - \alpha_k A\mathbf{p}_k, \quad \mathbf{p}_{k+1} = \mathbf{r}_{k+1} + \beta_k \mathbf{p}_k, \\ \mathbf{x}_{k+1}^* &= \mathbf{x}_k^* + \alpha_k^* \mathbf{p}_k^*, \quad \mathbf{r}_{k+1}^* = \mathbf{r}_k^* - \alpha_k^* A\mathbf{p}_k^*, \quad \mathbf{p}_{k+1}^* = \mathbf{r}_{k+1}^* + \beta_k^* \mathbf{p}_k^*. \end{aligned} \tag{A.4}$$

Using the bi-orthogonality properties and substitution, it is determined that the constants α and β are given as

$$\alpha_k = \alpha_k^* = \frac{(\mathbf{r}_k, \mathbf{r}_k^*)}{(A\mathbf{p}_k, \mathbf{p}_k^*)}, \quad \beta_k = \beta_k^* = \frac{(\mathbf{r}_{k+1}, \mathbf{r}_{k+1}^*)}{(\mathbf{r}_k, \mathbf{r}_k^*)}.$$

In Algorithm 7, these coefficients are computed and used to update the previous iterate. The solution of the dual problem $\mathbf{x}_k^*$ is not explicitly computed in Algorithm 7, since it is not directly needed in the computation of $\mathbf{x}_k$. However, if one would want to simultaneously solve the dual problem, only the line $\mathbf{x}_{k+1}^* = \mathbf{x}_k^* + \alpha_k \mathbf{p}_k^*$ has to be added.

Algorithm 7 Bi-conjugate gradient method (Bi-CG) algorithm

Choose initial guess $\mathbf{x}_0$
$\mathbf{r}_0 = \mathbf{f} - A\mathbf{x}_0$, choose $\mathbf{r}_0^*$ such that $(\mathbf{r}_0, \mathbf{r}_0^*) \neq 0$
$\mathbf{p}_0 = \mathbf{r}_0, \mathbf{p}_0^* = \mathbf{r}_0^*$
for $j = 1, 2, \ldots, k$ **do**
    $\alpha_j = \frac{(\mathbf{r}_j, \mathbf{r}_j^*)}{(A\mathbf{p}_j, \mathbf{p}_j^*)}$
    $\mathbf{x}_{j+1} = \mathbf{x}_j + \alpha_j \mathbf{p}_j$
    $\mathbf{r}_{j+1} = \mathbf{r}_j - \alpha_j A\mathbf{p}_j$
    $\mathbf{r}_{j+1}^* = \mathbf{r}_j^* - \alpha_j A^T \mathbf{p}_j^*$
    $\beta_j = \frac{(\mathbf{r}_{j+1}, \mathbf{r}_{j+1}^*)}{(\mathbf{r}_j, \mathbf{r}_j^*)}$
    $\mathbf{p}_{j+1} = \mathbf{r}_{j+1} + \beta_j \mathbf{p}_j$
    $\mathbf{p}_{j+1}^* = \mathbf{r}_{j+1}^* + \beta_j \mathbf{p}_j^*$
end for

A.3. The Bi-Conjugate Gradient stabilized (Bi-CGSTAB) Method

In the Bi-CG method, the coefficient matrix transpose A^T occurs in the computation of the coefficients α_k and β_k. However, in some applications this transpose is not available, motivating to consider transpose-free methods. Additionally, these transpose-free methods can converge faster for roughly the same computational costs. The bi-conjugate gradient stabilized (Bi-CGSTAB) method can be found in Algorithm 8 and is an iterative, transpose-free Krylov method that stabilizes the convergence behaviour seen in Bi-CG [60]. In Bi-CGSTAB, iterates are produced whose residual vectors are of the form

$$\mathbf{r}_k = \Upsilon_k(A)\phi(A)\mathbf{r}_0,$$

where $\phi_k(t)$ is the Bi-CG residual polynomial and $\Upsilon_k(t)$ the newly introduced stabilization polynomial. This stabilization polynomial aims to smooth the convergence behaviour of the Bi-CG algorithm and is defined recursively,

$$\Upsilon_k(t) = (1 - \omega_k t)\Upsilon_{k-1}(t).$$

Adding this stabilization polynomial to the identities of the original problem in Bi-CG as in (A.4), one gets the following residual and conjugate updates

$$\mathbf{r}_{k+1} = (I - \omega_k A)(\mathbf{r}_k - \alpha_k A\mathbf{p}_k), \quad \mathbf{p}_{k+1} = \mathbf{r}_{k+1} + \beta_k(I - \omega_k A)\mathbf{p}_k.$$

The computation of the coefficients α_k and β_k are rewritten algebraically such that direct application of A^T is not needed. The full derivation is given in [60] and in Chapter 7 of [49]. The parameter ω_k that occurs in the stabilization polynomial is chosen such that a steepest descent step is achieved in the residual direction.

The Bi-CGSTAB method has faster iteration times than generalized minimum residual (GMRES) and uses less memory. However, it is less robust than GMRES, especially when the (non-symmetric) coefficient matrix is ill-conditioned.

Algorithm 8 Bi-conjugate gradient stabilized method (Bi-CGSTAB) algorithm

Choose initial guess $\mathbf{x}_0$
$\mathbf{r}_0 = \mathbf{f} - A\mathbf{x}_0$, choose $\mathbf{r}_0^*$ such that $(\mathbf{r}_0, \mathbf{r}_0^*) \neq 0$
$\mathbf{p}_0 = \mathbf{r}_0$,
for $j = 1, 2, \ldots, k$ **do**
    $\alpha_j = \frac{(\mathbf{r}_j, \mathbf{r}_0^*)}{(A\mathbf{p}_j, \mathbf{r}_0^*)}$
    $\mathbf{s}_j = \mathbf{r}_j - \alpha_j A\mathbf{p}_j$
    $w_j = \frac{(A\mathbf{s}_j, \mathbf{s}_j)}{(A\mathbf{s}_j, A\mathbf{s}_j)}$
    $\mathbf{x}_{j+1} = \mathbf{x}_j + \alpha_j \mathbf{p}_j + \omega_j \mathbf{s}_j$
    $\mathbf{r}_{j+1} = \mathbf{s}_j - \omega_j A\mathbf{s}_j$
    $\beta_j = \frac{(\mathbf{r}_{j+1}, \mathbf{r}_0^*)}{(\mathbf{r}_j, \mathbf{r}_0^*)} \times \frac{\alpha_j}{\omega_j}$
    $\mathbf{p}_{j+1} = \mathbf{r}_{j+1} + \beta_j(\mathbf{p}_j - \omega_j A\mathbf{p}_j)$
end for

B

Energy Network Data and Graphs

This chapter contains the graph representation of the test networks considered in this research. Section B.1 contains the directed graphs of the GasLib datasets, Section B.2 contains the undirected graphs of the PandaPower datasets.

B.1. GasLib

This section contains a selection of the GasLib datasets graph network representations used in this research. The GasLib datasets are elaborated on further in Section 8.1. In the graph network representations a blue node represents a source, a red node represents a sink, and a black node represents a junction. The green link is a gas pipe (high and low pressure), a yellow link a compressor station and a pink link a resistor, short pipe or (control) valve.

GasLib-11 Network Graph

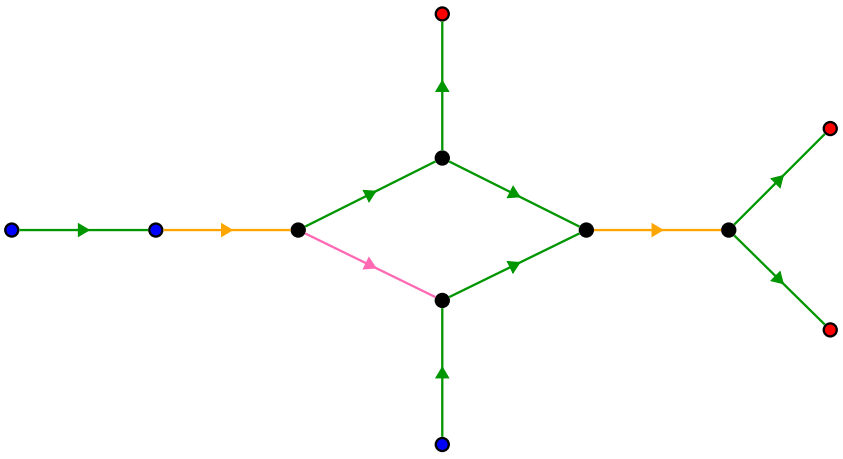


Figure B.1: Network graph of the GasLib-11 dataset. A blue node represents a source, a red node a sink and a black node a junction.

GasLib-24 Network Graph

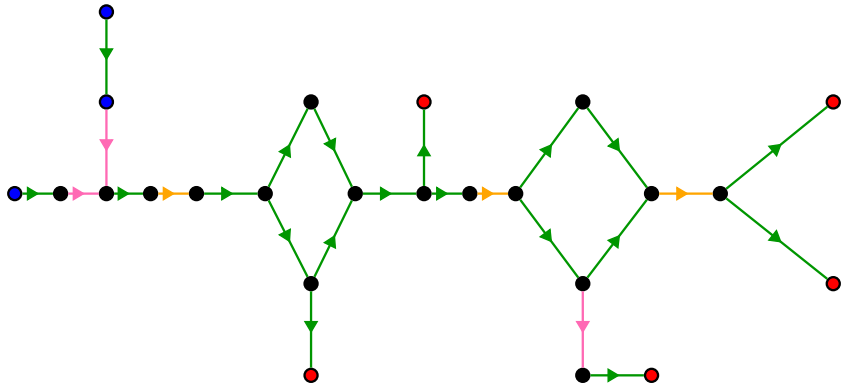


Figure B.2: Network graph of the GasLib-24 dataset. A blue node represents a source, a red node a sink and a black node a junction.

GasLib-40 Network Graph

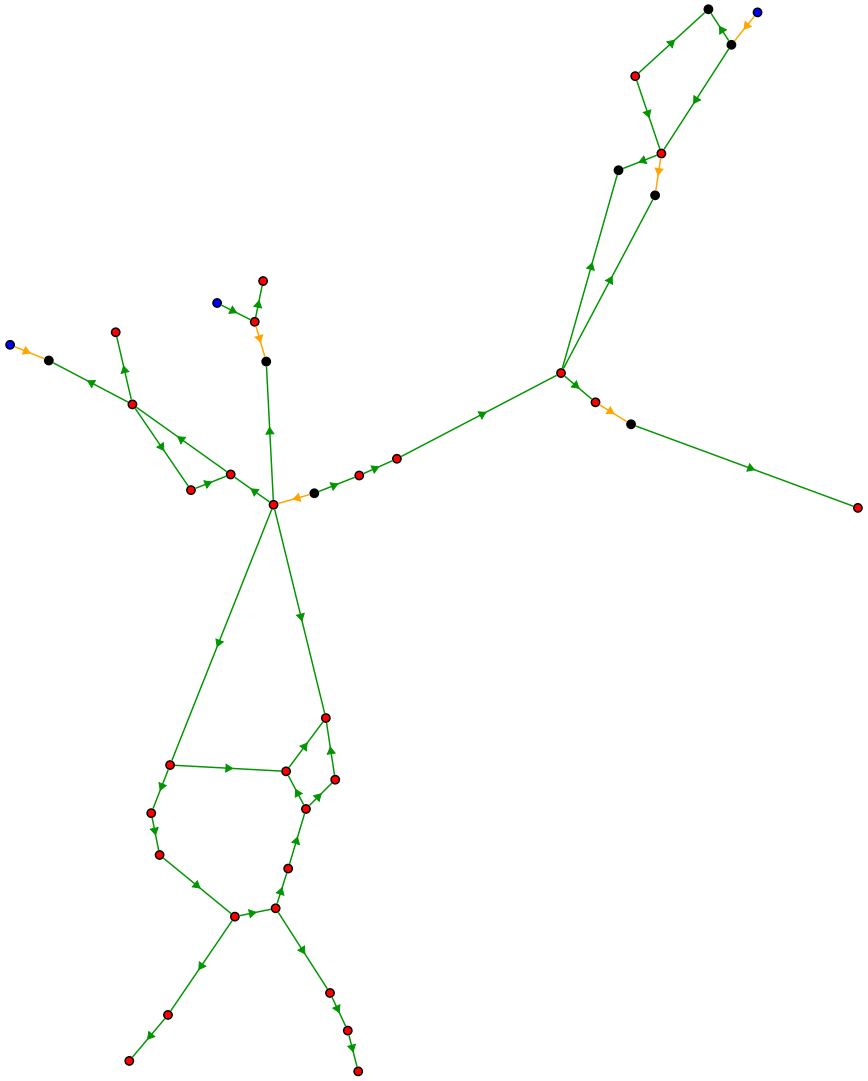


Figure B.3: Network graph of the GasLib-40 dataset. A blue node represents a source, a red node a sink and a black node a junction.

GasLib-135 Network Graph

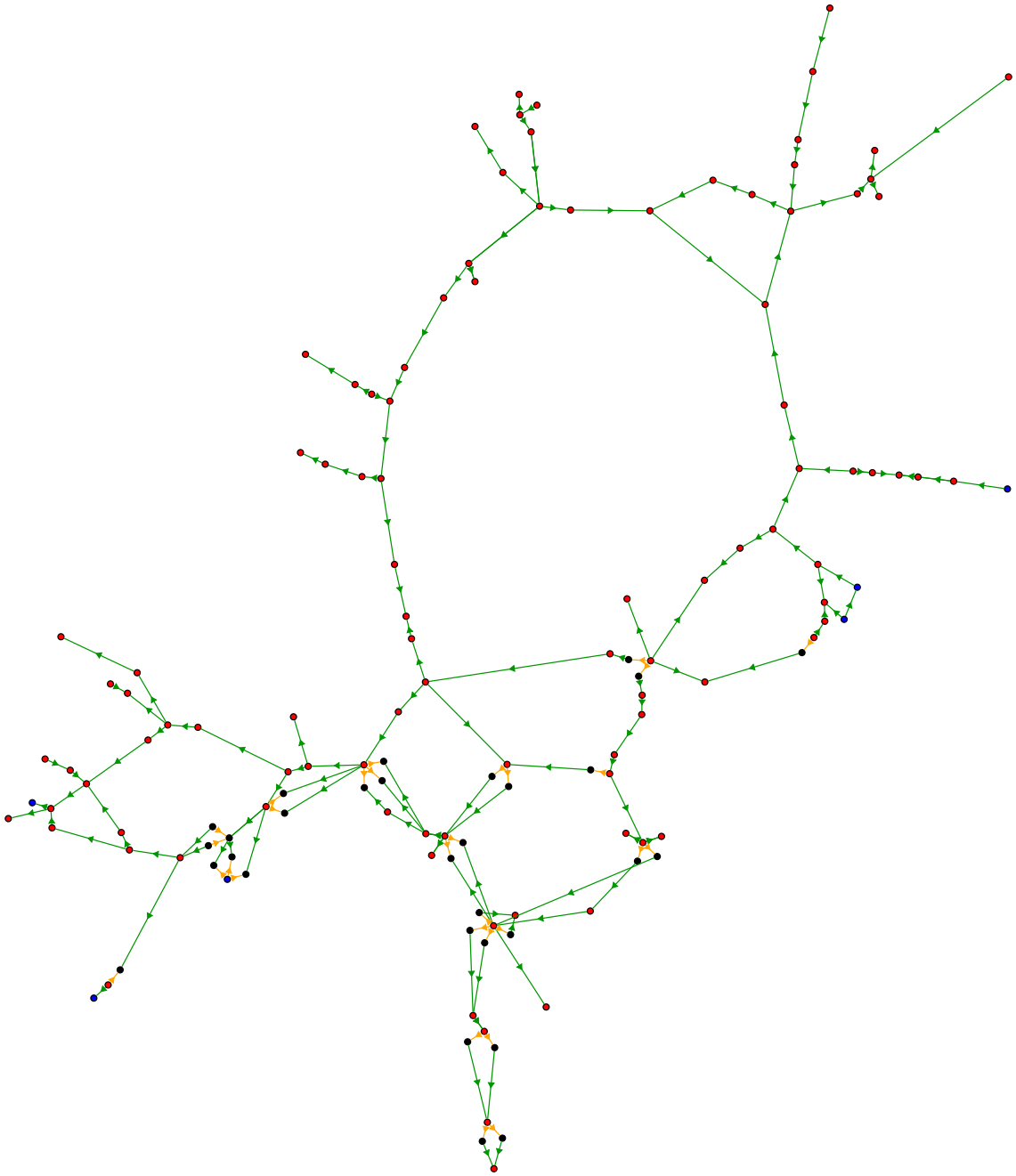


Figure B.4: Network graph of the GasLib-135 dataset. A blue node represents a source, a red node a sink and a black node a junction. A compressor station is indicated with an orange link colour.

GasLib-582 Network Graph

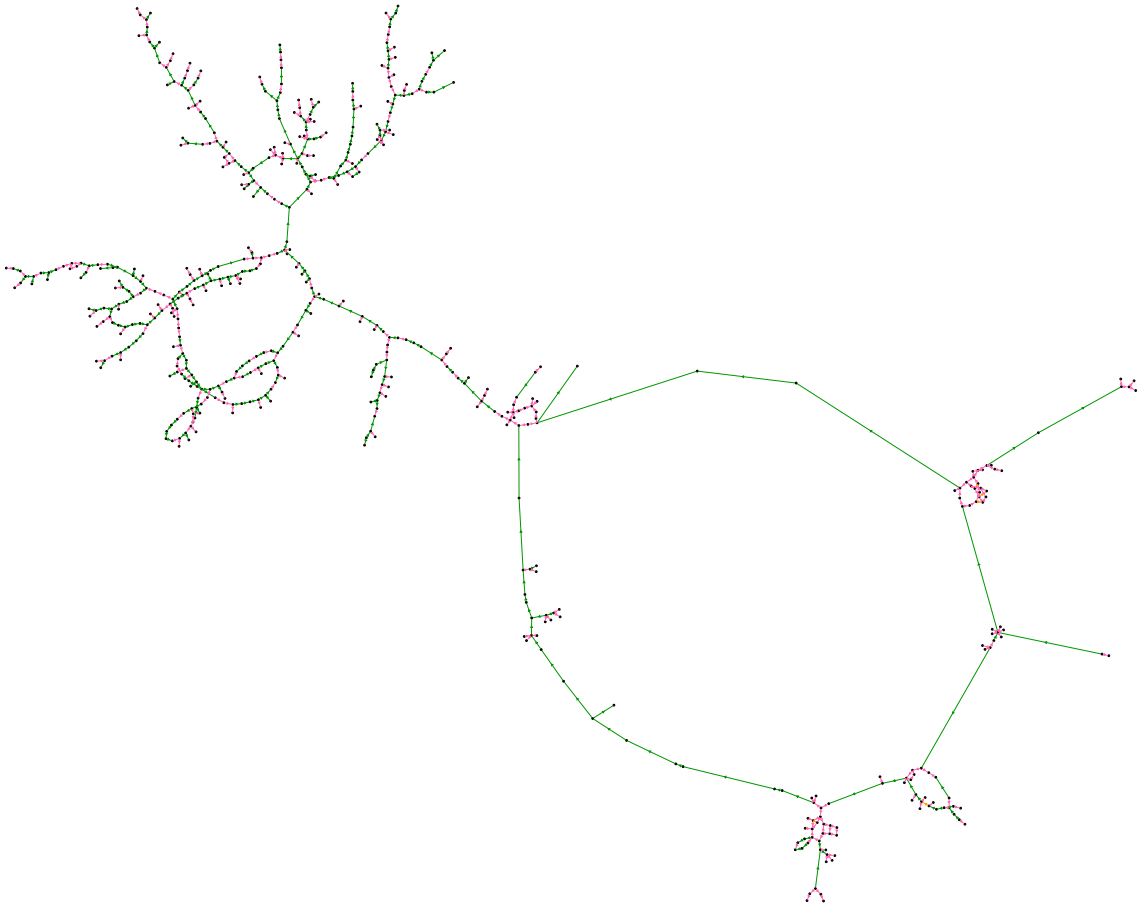


Figure B.5: Network graph of the GasLib-582 dataset. A blue node represents a source, a red node a sink and a black node a junction. A compressor station is indicated with an orange link colour.

B.2. PandaPower

This section contains a selection of the PandaPower datasets graph network representations used in this research. The PandaPower datasets are elaborated on further in Section 8.1. The node locations and link lengths are not pre-determined. As a result, the graph representation is schematic, and not realistic to scale. In the graph network representations a blue node represents a PQ node, a red node represents a PV node, and a green node represents slack node.

Case9 Network Graph

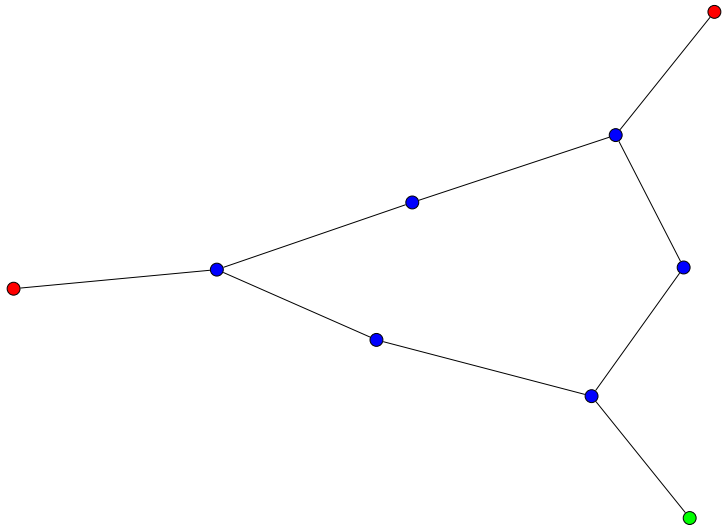


Figure B.6: Network graph of the case9 dataset. A blue node represents a PQ node, a red node a PV node, and a green node the slack node.

Case14 Network Graph

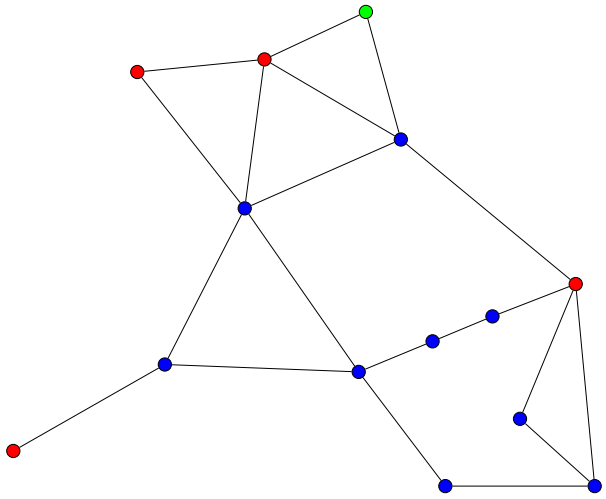


Figure B.7: Network graph of the case14 dataset. A blue node represents a PQ node, a red node a PV node, and a green node the slack node.

Case30 Network Graph

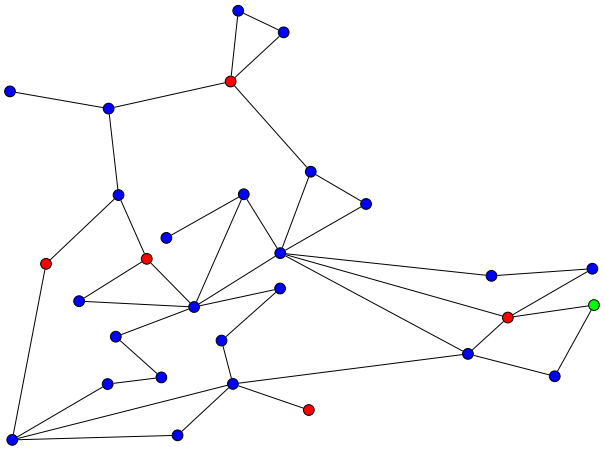


Figure B.8: Network graph of the case30 dataset. A blue node represents a PQ node, a red node a PV node, and a green node the slack node.

Case89pegase Network Graph

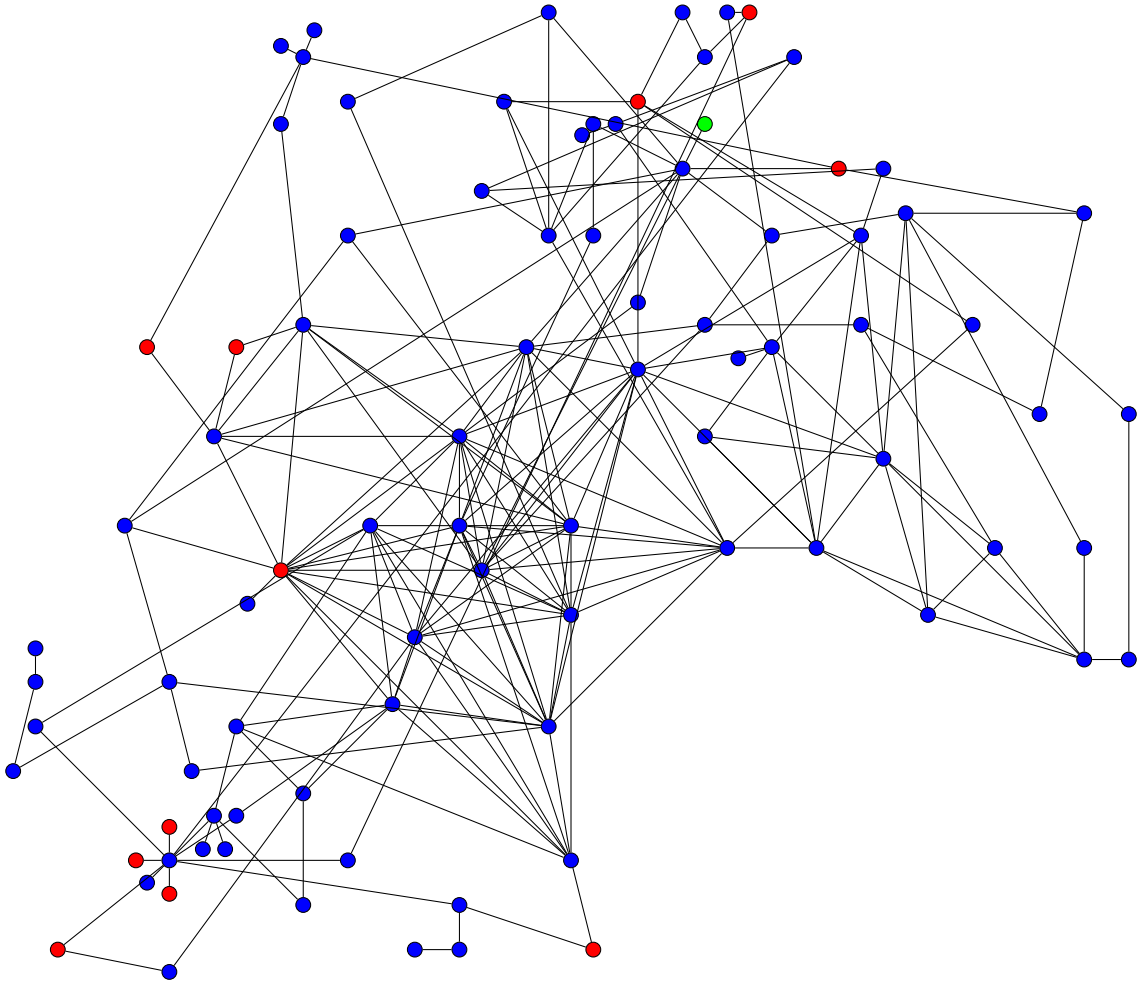


Figure B.9: Network graph of the case89pegase dataset. A blue node represents a PQ node, a red node a PV node, and a green node the slack node.

C

Coupled Energy Network Data

This chapter contains an overview of the coupling possibilities of the GasLib and PandaPower datasets when constructing a coupled energy network. The restrictions on available nodes for coupling power-to-gas (P2G) and gas-fired generator (GFG) units is described in more detail in Section 8.2.2.

Table C.1: Maximum number of P2G and GFG coupling units that can be connected to a gas network in the GasLib database.

GasLib Network	Max. P2G units	Max. GFG units
GasLib-11	1	3
GasLib-24	2	5
GasLib-40	2	29
GasLib-135	5	99
GasLib-582	11	50
GasLib-2607	30	230
GasLib-4197	39	747

Table C.2: Maximum number of P2G and GFG coupling units that can be connected to an electrical network in the PandaPower database.

Pandapower Network	Max. P2G units	Max. GFG units
case4gs	2	0
case5	1	0
case6ww	3	0
case9	3	0
case14	7	0
case24_ieee_rts	8	0
GBreducednetwork	5	0
case30	17	0
case_ieee30	16	0
case33bw	32	0
case39	19	0
case57	32	0
case89pegase	**24**	**4**
case118	48	0
case145	7	0
iceland	45	0
case_illinois200	105	0
case300	**133**	**8**
case1354pegase	6	0
case1888rte	**867**	**64**
GBnetwork	**229**	**35**
case2848rte	**1307**	**160**
case2869pegase	**33**	**84**
case3120sp	2035	0
case6470rte	**3209**	**447**
case6495rte	**3138**	**327**
case6515rte	**3167**	**311**
case9241pegase	**89**	**91**

D

Test Networks Solver Configuration Data

This chapter contains an overview of the performance of various linear solver configurations for the test networks of the GasLib and PandaPower datasets. The total solution time was measures by taking the average of 100 solution cycles. Table D.1 and Figure D.1 present the solution times, final residuals and iteration counts of the GasLib datasets. Similarly, Table D.2 and Figure D.2 present the solution times, final residuals and iteration counts of the PandaPower datasets.

Table D.1: Combined performance overview comparing solution times, final residuals, and solver-specific parameters across different linear solver configurations.

Network	Metric / Param.	Direct (LU)	GMRES (Un-pre.)	GMRES + ILU(t)	GMRES + RAS
GasLib-11	NR its.	4	4	4	4
	Tot. linear its.	-	84	16	20
	Final Residual	$2.788751 \cdot 10^{-12}$	$3.090020 \cdot 10^{-12}$	$3.090019 \cdot 10^{-12}$	$3.090013 \cdot 10^{-12}$
	Solution Time	$7.6 \cdot 10^{-3}$ sec.	$3.3 \cdot 10^{-2}$ sec.	$1.3 \cdot 10^{-2}$ sec.	$2.1 \cdot 10^{-2}$ sec.
	Hyperparam.	-	-	$t = 1$	4 sub / 2 ovlp.
GasLib-24	NR its.	3	3	3	3
	Tot. linear its.	-	144	3	24
	Final Residual	$2.994329 \cdot 10^{-10}$	$2.994329 \cdot 10^{-10}$	$2.994333 \cdot 10^{-10}$	$7.492076 \cdot 10^{-8}$
	Solution Time	$6.4 \cdot 10^{-3}$ sec.	$4.4 \cdot 10^{-2}$ sec.	$8.4 \cdot 10^{-3}$ sec.	$2.2 \cdot 10^{-2}$ sec.
	Hyperparam.	-	-	$t = 4$	4 sub / 2 ovlp.
GasLib-40	NR its.	5	5	5	5
	Tot. linear its.	-	420	5	55
	Final Residual	$7.492161 \cdot 10^{-8}$	$7.492161 \cdot 10^{-8}$	$7.492161 \cdot 10^{-8}$	$7.492076 \cdot 10^{-8}$
	Solution Time	$1.1 \cdot 10^{-2}$ sec.	$1.9 \cdot 10^{-1}$ sec.	$1.5 \cdot 10^{-2}$ sec.	$3.9 \cdot 10^{-2}$ sec.
	Hyperparam.	-	-	$t = 5$	4 sub / 2 ovlp.
GasLib-135	NR its.	9	9	9	9
	Tot. linear its.	-	2,736	10	169
	Final Residual	$3.477431 \cdot 10^{-7}$	$3.477431 \cdot 10^{-7}$	$3.477431 \cdot 10^{-7}$	$3.477431 \cdot 10^{-7}$
	Solution Time	$3.2 \cdot 10^{-2}$ sec.	$3.6 \cdot 10^{0}$ sec.	$4.2 \cdot 10^{-2}$ sec.	$1.3 \cdot 10^{-1}$ sec.
	Hyperparam.	-	-	$t = 5$	5 sub / 3 ovlp.
GasLib-582	NR its.	14	15	14	14
	Tot. linear its.	-	17,849	56	93
	Final Residual	$2.428902 \cdot 10^{-7}$	$1.504489 \cdot 10^{-6}$	$2.428902 \cdot 10^{-7}$	$2.428901 \cdot 10^{-7}$
	Solution Time	$1.3 \cdot 10^{-1}$ sec.	$9.3 \cdot 10^{1}$ sec.	$1.7 \cdot 10^{-1}$ sec.	$2.9 \cdot 10^{-1}$ sec.
	Hyperparam.	-	-	$t = 5$	3 sub / 2 ovlp.
GasLib-2607	NR its.	-	-	-	16
	Tot. linear its.	-	-	-	700
	Final Residual	-	-	-	$3.964603 \cdot 10^{-7}$
	Solution Time	-	-	-	$3.1 \cdot 10^{0}$ sec.
	Hyperparam.	-	-	-	4 sub / 2 ovlp.
GasLib-4197	NR its.	13	18	13	13
	Tot. linear its.	-	155,908	149	543
	Final Residual	$8.509677 \cdot 10^{-7}$	$8.132072 \cdot 10^{-7}$	$8.509708 \cdot 10^{-7}$	$8.509359 \cdot 10^{-7}$
	Solution Time	$3.7 \cdot 10^{0}$ sec.	$7.2 \cdot 10^{3}$ sec.	$1.2 \cdot 10^{0}$ sec.	$3.7 \cdot 10^{0}$ sec.
	Hyperparam.	-	-	$t = 5$	4 sub / 2 ovlp.

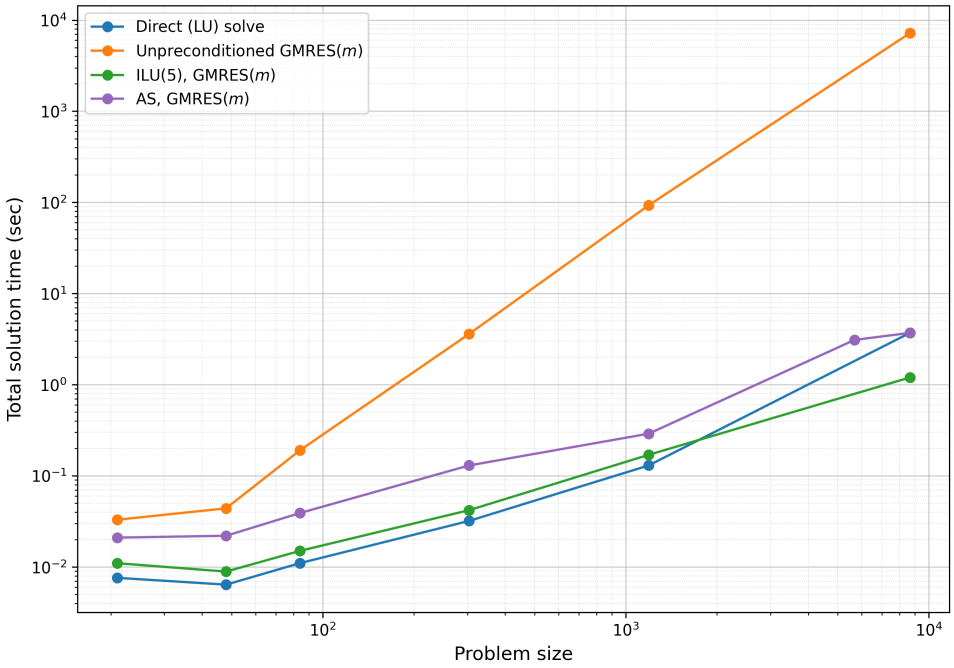


Figure D.1: Solution times as a function of the problem size for the GasLib datasets, with different linear solver configurations.

Table D.2: Combined performance overview comparing solution times, final residuals, and solver-specific parameters across different linear solver configurations for PandaPower datasets.

Network	Metric / Param.	Direct (LU)	GMRES (Un-pre.)	GMRES + ILU(t)	GMRES + RAS
case9	NR its.	3	3	3	3
	Tot. linear its.	-	42	6	9
	Final Residual	$8.853004 \cdot 10^{-7}$	$8.853004 \cdot 10^{-7}$	$8.853004 \cdot 10^{-7}$	$8.853004 \cdot 10^{-7}$
	Solution Time	$7.7 \cdot 10^{-3}$ sec.	$2.0 \cdot 10^{-2}$ sec.	$1.1 \cdot 10^{-2}$ sec.	$1.5 \cdot 10^{-2}$ sec.
	Hyperparam.	-	-	$t = 1$	4 sub / 2 ovlp.
case14	NR its.	3	3	3	3
	Tot. linear its.	-	66	24	21
	Final Residual	$1.023082 \cdot 10^{-7}$	$1.023082 \cdot 10^{-7}$	$1.023082 \cdot 10^{-7}$	$1.023082 \cdot 10^{-7}$
	Solution Time	$9.0 \cdot 10^{-3}$ sec.	$2.4 \cdot 10^{-2}$ sec.	$1.6 \cdot 10^{-2}$ sec.	$1.8 \cdot 10^{-2}$ sec.
	Hyperparam.	-	-	$t = 1$	4 sub / 2 ovlp.
case30	NR its.	3	3	3	3
	Tot. linear its.	-	159	78	39
	Final Residual	$1.603518 \cdot 10^{-9}$	$1.603485 \cdot 10^{-9}$	$1.603475 \cdot 10^{-9}$	$1.603474 \cdot 10^{-9}$
	Solution Time	$1.0 \cdot 10^{-2}$ sec.	$5.4 \cdot 10^{-2}$ sec.	$2.8 \cdot 10^{-2}$ sec.	$2.8 \cdot 10^{-2}$ sec.
	Hyperparam.	-	-	$t = 1$	4 sub / 2 ovlp.
case89pegase	NR its.	4	4	4	4
	Tot. linear its.	-	519	356	64
	Final Residual	$2.726046 \cdot 10^{-9}$	$2.726229 \cdot 10^{-9}$	$2.726132 \cdot 10^{-9}$	$2.726049 \cdot 10^{-9}$
	Solution Time	$2.2 \cdot 10^{-2}$ sec.	$3.5 \cdot 10^{-1}$ sec.	$1.7 \cdot 10^{-1}$ sec.	$5.6 \cdot 10^{-2}$ sec.
	Hyperparam.	-	-	$t = 1$	4 sub / 2 ovlp.
case300	NR its.	5	5	5	5
	Tot. linear its.	-	2,338	475	123
	Final Residual	$3.578654 \cdot 10^{-12}$	$4.487582 \cdot 10^{-12}$	$3.969428 \cdot 10^{-12}$	$3.623571 \cdot 10^{-12}$
	Solution Time	$4.1 \cdot 10^{-2}$ sec.	$3.6 \cdot 10^{0}$ sec.	$3.0 \cdot 10^{-1}$ sec.	$1.3 \cdot 10^{-1}$ sec.
	Hyperparam.	-	-	$t = 1$	4 sub / 2 ovlp.
case2869pegase	NR its.	5	5	5	5
	Tot. linear its.	-	8,347	2,483	139
	Final Residual	$5.916868 \cdot 10^{-11}$	$6.114068 \cdot 10^{-11}$	$6.202309 \cdot 10^{-11}$	$5.914377 \cdot 10^{-11}$
	Solution Time	$3.7 \cdot 10^{-1}$ sec.	$6.9 \cdot 10^{1}$ sec.	$5.3 \cdot 10^{0}$ sec.	$1.1 \cdot 10^{0}$ sec.
	Hyperparam.	-	-	$t = 1$	4 sub / 2 ovlp.
case6495rte	NR its.	6	6	6	6
	Tot. linear its.	-	20,084	4,983	176
	Final Residual	$1.952497 \cdot 10^{-8}$	$1.952540 \cdot 10^{-8}$	$1.952563 \cdot 10^{-8}$	$1.952530 \cdot 10^{-8}$
	Solution Time	$8.7 \cdot 10^{-1}$ sec.	$2.3 \cdot 10^{3}$ sec.	$6.2 \cdot 10^{1}$ sec.	$4.0 \cdot 10^{0}$ sec.
	Hyperparam.	-	-	$t = 1$	4 sub / 2 ovlp.
case6515rte	NR its.	10	10	10	10
	Tot. linear its.	-	35,185	8,355	175
	Final Residual	$6.744816 \cdot 10^{-11}$	$6.444298 \cdot 10^{-11}$	$7.544585 \cdot 10^{-11}$	$6.699047 \cdot 10^{-11}$
	Solution Time	$1.4 \cdot 10^{0}$ sec.	$3.9 \cdot 10^{3}$ sec.	$1.2 \cdot 10^{2}$ sec.	$5.1 \cdot 10^{0}$ sec.
	Hyperparam.	-	-	$t = 1$	4 sub / 2 ovlp.
case9241pegase	NR its.	6	6	6	6
	Tot. linear its.	-	30,953	43	247
	Final Residual	$5.769165 \cdot 10^{-11}$	$5.664056 \cdot 10^{-11}$	$5.597867 \cdot 10^{-11}$	$5.748121 \cdot 10^{-11}$
	Solution Time	$1.6 \cdot 10^{0}$ sec.	$6.1 \cdot 10^{3}$ sec.	$1.9 \cdot 10^{0}$ sec.	$7.0 \cdot 10^{0}$ sec.
	Hyperparam.	-	-	$t = 4$	4 sub / 2 ovlp.

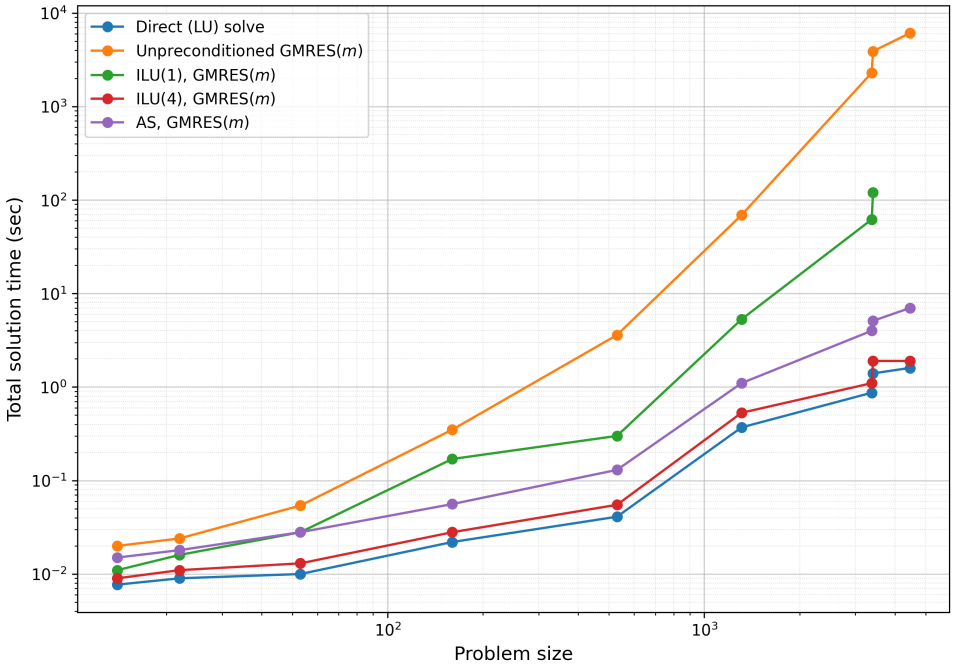


Figure D.2: Solution times as a function of the problem size for the PandaPower datasets, with different linear solver configurations.