Introduction to a high-performance multi-period optimal power flow solver

Salman Zaferanlouei

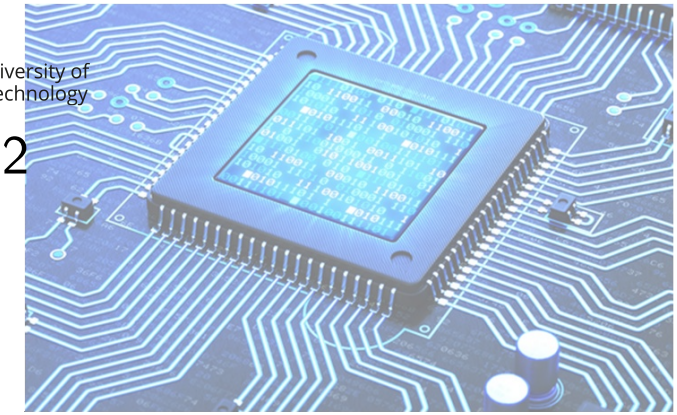
Winter School Workshop 2022

Quality Hotel Skifer

Oppdal

NTNU | Norwegian University of Science and Technology

March 3, 2022



The presentation goal

Purpose: To discuss how to solve multi-period optimal power flow fast; **Large-Scale simulation with respect to time and space.**

Phase I: Background and Motivation

Phase II: Power Flow and Optimal Power Flow

Phase III: Solution Method

Phase IV: Speed up

Phase V: Future Work

Presentation Time:	15-20 min

Phase I:

Background and Motivation

IBM 7090

Figure 1: [Photo: NASA]

Items

It took up a whole room and it was state of the art when it came out in 1960!

Figure 2: NASA

Items

It took up a whole room and it was state of the art when it came out in 1960!

It costs 2.9 million dollars u.s. at the time which is a lot more now!

Figure 2: NASA

Items

It took up a whole room and it was state of the art when it came out in 1960!

It costs 2.9 million dollars u.s. at the time which is a lot more now!

It could perform a hunderd thousand floating point operation per second (we call it flops)

Figure 2: NASA

Items

It took up a whole room and it was state of the art when it came out in 1960!

It costs 2.9 million dollars u.s. at the time which is a lot more now!

It could perform a hunderd thousand floating point operation per second (we call it flops)

🤔

So Does that sound fast? pretty fast?

Figure 2: NASA

Items

It took up a whole room and it was state of the art when it came out in 1960!

It costs 2.9 million dollars u.s. at the time which is a lot more now!

It could perform a hunderd thousand floating point operation per second (we call it flops)

😮

a first generation Raspberry Pi; The little thing that everybody builds their toy projects with these days was over 40 times faster!

Figure 2: NASA

How things got better?

Make Bigger, Faster computers.

For a long time we just basically changed the hardware to make computers better by stuffing more transistors onto a chip and running instruction through a system more quickly. We built bigger and faster computers as they get cheaper, we can through more of them in a given problem.

CPU

Gordon Moore has quantified this growth in 1965 when he coined Moore's law! Moore's law says that the number of components on a chip would double every 18 months.
Doubling number of components every 18 months meant the computers got twice as powerful while sitting in the same space.

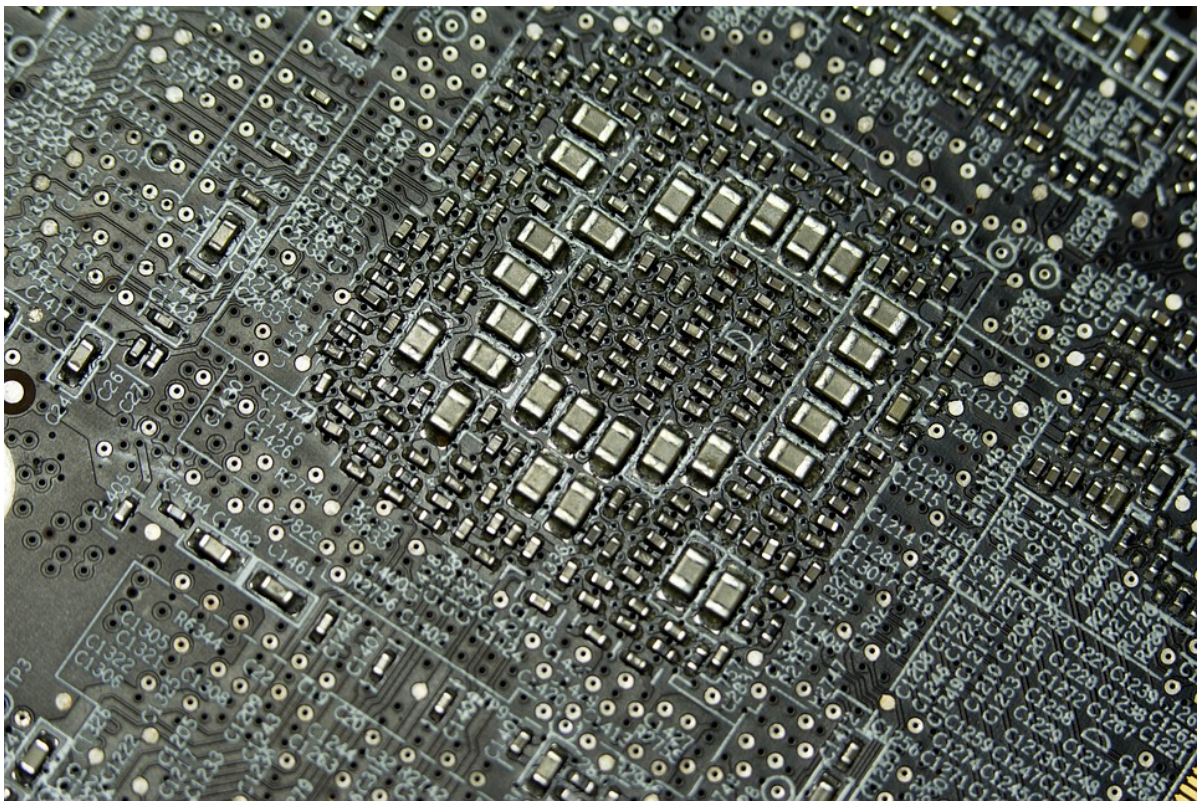


Figure 3: [Image bia pixabay]

Problems/arguments

However there is a few problems with this:

i. The law of Physics says that eventually you can't make a component any smaller than a certain size,

Problems/arguments

However there is a few problems with this:

i. The law of Physics says that eventually you can't make a component any smaller than a certain size,

ii. It gets more expensive to build smaller chips as you get close to those limits.

Problems/arguments

However there is a few problems with this:

i. The law of Physics says that eventually you can't make a component any smaller than a certain size,

ii. It gets more expensive to build smaller chips as you get close to those limits.

iii. But customers, the people who use and buy computers and keep the computer manufacurers in business do not care!

Problems/arguments

However there is a few problems with this:

i. The law of Physics says that eventually you can't make a component any smaller than a certain size,

ii. It gets more expensive to build smaller chips as you get close to those limits.

iii. But customers, the people who use and buy computers and keep the computer manufacurers in business do not care!

iv. Programmers keep writing bigger and more complex programs and as long as the new computers are speeding up more rapidly than your software is growing, things seem to be improving!

intr.

When a program needs a bigger and newer computer to run it what happens to the old one? It might handed down to someone or resold or it might get thrown out,
And this is not good for the planet!
Also, having code only works well on the newest devices is not great for the people who can not afford the sate of the art!
If you have a business, you have a problem, because you have to get your code or your website on whatever people are using, rather than what you wish they were using, So if your program is so slow, they might not be your custormer for a very long,

Intro

People want their computers keep getting better But, Just making the hardware faster is not nearly enough anymore!
And Throwing more hardware at a problem can get VERY expensive.
It seems crutial to modify softwares at the same time.

Electricity Grid

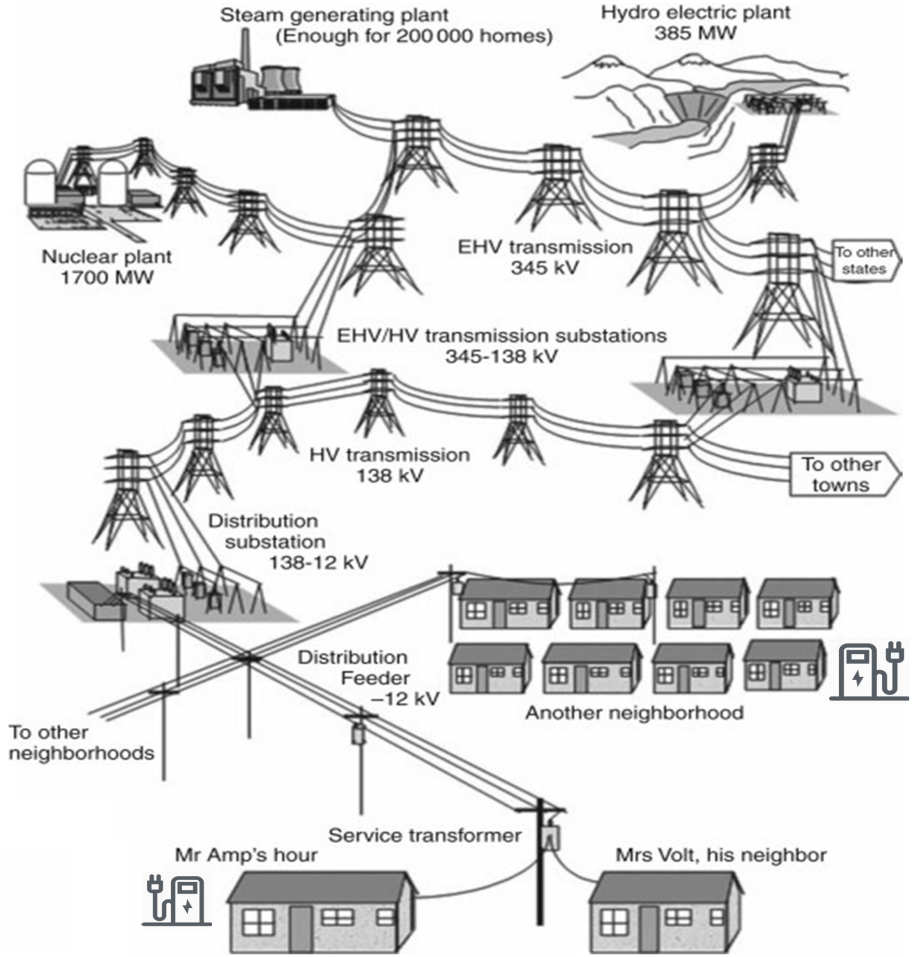


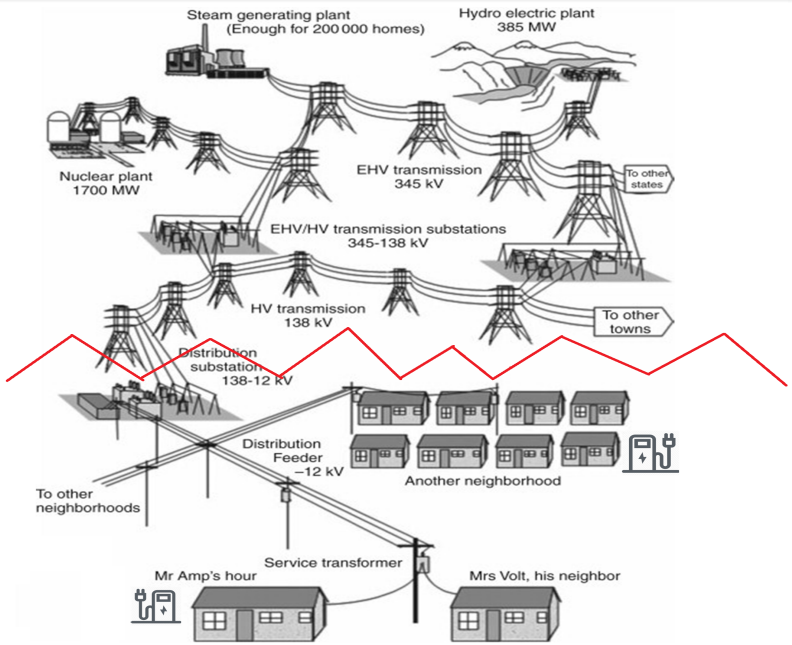
Figure 6: [www.sciencedirect.com/science/article/pii/B9781845697846500019]

Sustainability/Green shift/CO_2 reduction

For many reasons power electricity grid is facing decentralization

Phasing out coal (carbon-heavy sources of production) and nuclear power plants

This chain between large power producers and consumers is weakened.

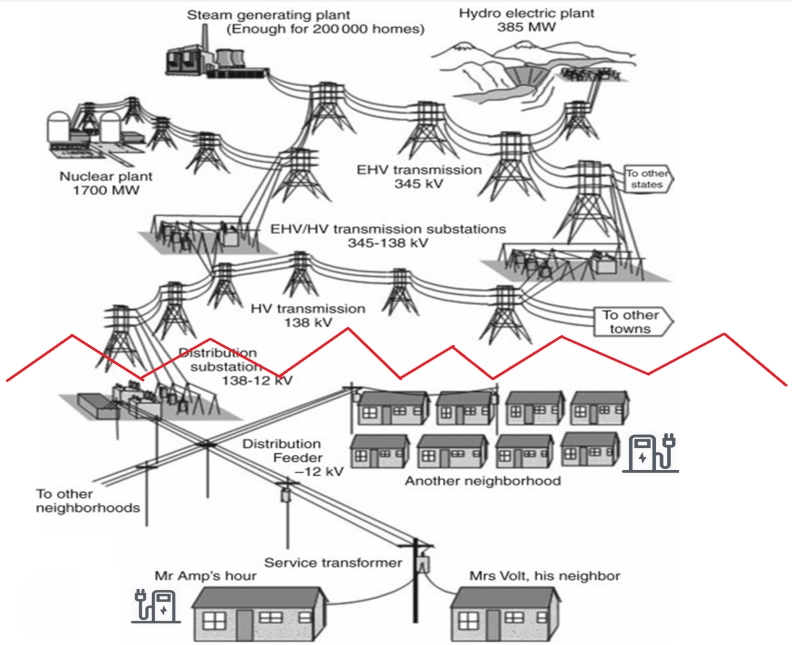


Sustainability/Green shift/CO_2 reduction

For many reasons power electricity grid is facing decentralization

- Phasing out coal (carbon-heavy sources of production) and nuclear power plants
- Increase penetration of solar and wind production

This chain between large power producers and consumers is weakened.



Background: cont.

Items

Green shift in electricity systems is needed for the reduction of CO_2 emissions

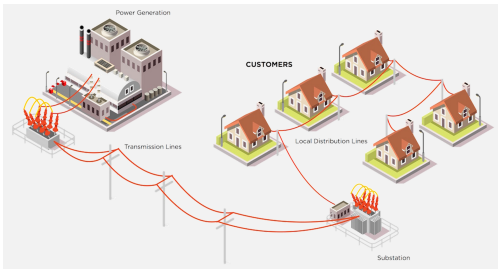


Figure 7: A Typical Power System [Rochester Gas & Electricity]

Background: cont.

Items

Green shift in electricity systems is needed for the reduction of CO_2 emissions

Integration of Distributed Energy Resources (DER) is a huge challenge

Note

DER includes Renewable Energy, Energy Storage, Electric Vehicles and Flexible Demand

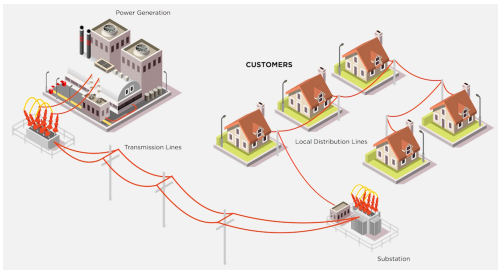


Figure 7: A Typical Power System [Rochester Gas & Electricity]

Background: cont.

Items

Green shift in electricity systems is needed for the reduction of CO_2 emissions

Integration of Distributed Energy Resources (DER) is a huge challenge

Grid companies must be able to analyse the impacts of DER

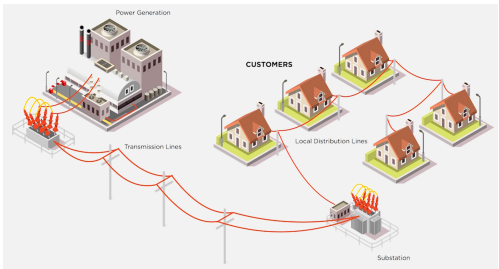


Figure 7: A Typical Power System [Rochester Gas & Electricity]

Background: cont.

Items

Green shift in electricity systems is needed for the reduction of CO_2 emissions

Integration of Distributed Energy Resources (DER) is a huge challenge

Grid companies must be able to analyse the impacts of DER

Note

Optimal Power Flow (OPF)solvers are essential

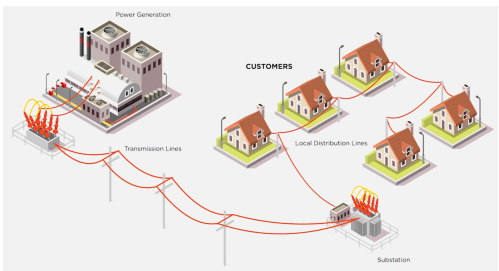


Figure 7: A Typical Power System [Rochester Gas & Electricity]

Background: cont.

Items

Green shift in electricity systems is needed for the reduction of CO_2 emissions

Integration of Distributed Energy Resources (DER) is a huge challenge

Grid companies must be able to analyse the impacts of DER

The inegration of DER in smart grids calls for **much more sophisticated solvers** for OPF

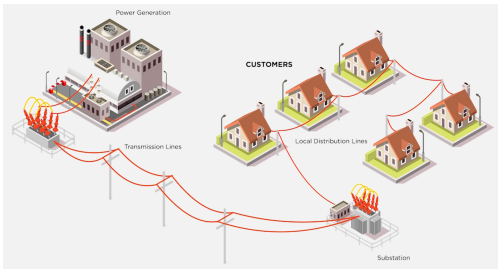


Figure 7: A Typical Power System [Rochester Gas & Electricity]

Challenges in the planning and operation of the grid

Planning: Optimizing the right type, size and timing of new grid investments

- Local generation (e.g. PV) and increased load (e.g. EVs) can be located in areas where the grid is weak
- Energy storage and demand flexibility are alternatives to grid reinforcements

Challenges in the planning and operation of the grid

Planning: Optimizing the right type, size and timing of new grid investments

- Local generation (e.g. PV) and increased load (e.g. EVs) can be located in areas where the grid is weak
- Energy storage and demand flexibility are alternatives to grid reinforcements

Operation: Optimize the use of controllable assets such as energy storage and flexible demand to secure, reliable and economic operation of the distribution grids. This means:

- Making the right use of Demand Response
- being able to value the use of end-user flexibility for local or system-wide grid services
- Simulating and optimizing the grids in the presence of **future local markets for energy and flexibility**

Limitations of traditional grid operation and planning

Notes

Classical single-period OPF does not offer a possibility for optimal operational scheduling of storage and flexible demand

Limitations of traditional grid operation and planning

Notes

Classical single-period OPF does not offer a possibility for optimal operational scheduling of storage and flexible demand

We therefore aim to develop the foundations for a new generation of Multi-Period OPF (MPOPF) solvers

i. Solves the OPF problem over several coupled time-steps

ii. Computation time is an issue when using both commercial or free optimization solvers

Limitations of traditional grid operation and planning

Notes

Classical single-period OPF does not offer a possibility for optimal operational scheduling of storage and flexible demand

We therefore aim to develop the foundations for a new generation of Multi-Period OPF (MPOPF) solvers

i. Solves the OPF problem over several coupled time-steps
ii. Computation time is an issue when using both commercial or free optimization solvers

MPOPF is an extremely challenging scientific task:

i. Nonlinearity
ii. Large-scale problem with respect with to time and space
iii. Involves stochastic generations and load

Limitations of traditional grid operation and planning

Notes

Classical single-period OPF does not offer a possibility for optimal operational scheduling of storage and flexible demand

We therefore aim to develop the foundations for a new generation of Multi-Period OPF (MPOPF) solvers

i. Solves the OPF problem over several coupled time-steps
ii. Computation time is an issue when using both commercial or free optimization solvers

MPOPF is an extremely challenging scientific task:

i. Nonlinearity
ii. Large-scale problem with respect with to time and space
iii. Involves stochastic generations and load

Hardware is reaching its limit with respect to CPU clock speed

Solution:

High-Performance Solver [1-2]

Algorithmic design tailored to the conventional OPF algorithms speed-up the solution proposal

Prototype model shows convincing results for real-sized system with distributed renewables, storages and EVs

i. A high-performance and memory-efficient sparse algorithm

ii. Utilizing the structure of the underlying mathematical formulation

1. S. Zaferanlouei, H. Farahmand, V. V. Vadlamudi, M. Korpås,BATTPOWER Toolbox: Memory-Efficient and High-Performance Multi-Period AC Optimal Power Flow Solver, IEEE Transactions on Power Systems, Jan. 16th, 2021.

2. S. Zaferanlouei, et al., BATTPOWER Application: Large-Scale Integration of EVs in an Active Distribution Grid —A Norwegian Case Study, Under review in the journal of EPSR

Solution:

High-Performance Solver [1-2]

Algorithmic design tailored to the conventional OPF algorithms speed-up the solution proposal

Prototype model shows convincing results for real-sized system with distributed renewables, storages and EVs

i. A high-performance and memory-efficient sparse algorithm

ii. Utilizing the structure of the underlying mathematical formulation

Benefits

Optimal utilization of **stored energy** and **flexibility** where and when it creates the highest value for the system

1. S. Zaferanlouei, H. Farahmand, V. V. Vadlamudi, M. Korpås,BATTPOWER Toolbox: Memory-Efficient and High-Performance Multi-Period AC Optimal Power Flow Solver, IEEE Transactions on Power Systems, Jan. 16th, 2021.

2. S. Zaferanlouei, et al., BATTPOWER Application: Large-Scale Integration of EVs in an Active Distribution Grid —A Norwegian Case Study, Under review in the journal of EPSR

Solution:

High-Performance Solver [1-2]

Algorithmic design tailored to the conventional OPF algorithms speed-up the solution proposal

Prototype model shows convincing results for real-sized system with distributed renewables, storages and EVs

i. A high-performance and memory-efficient sparse algorithm

ii. Utilizing the structure of the underlying mathematical formulation

Benefits

Optimal utilization of **stored energy** and **flexibility** where and when it creates the highest value for the system

Can be used for grid planning, grid operation and local markets

1. S. Zaferanlouei, H. Farahmand, V. V. Vadlamudi, M. Korpås,BATTPOWER Toolbox: Memory-Efficient and High-Performance Multi-Period AC Optimal Power Flow Solver, IEEE Transactions on Power Systems, Jan. 16th, 2021.

2. S. Zaferanlouei, et al., BATTPOWER Application: Large-Scale Integration of EVs in an Active Distribution Grid —A Norwegian Case Study, Under review in the journal of EPSR

Power System— Today

Power System— Future

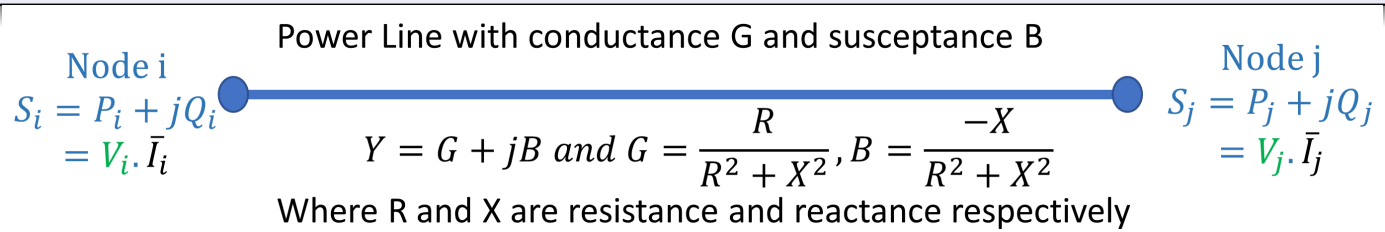
Phase II:

Power Flow and Optimal Power Flow

Power Flow Equations

Source of Non-linearity

Nonlinear relationship between the voltage phasors and the power injections



$$
\begin{aligned}
P_i + jQ_i &= V_i.\bar{I}_i \\
&= V_i.(\overline{\mathbf{Y}}_i.\overline{V}) \\
&= V_i.(\mathbf{G}_i - j\mathbf{B}_i)\overline{V} \\
|V_i|^2 &= V_i.\overline{V}_i
\end{aligned}
$$

"Power Flow Equations"

"Load Flow Equations"

$\mathbf{Y}$ admittance matrix

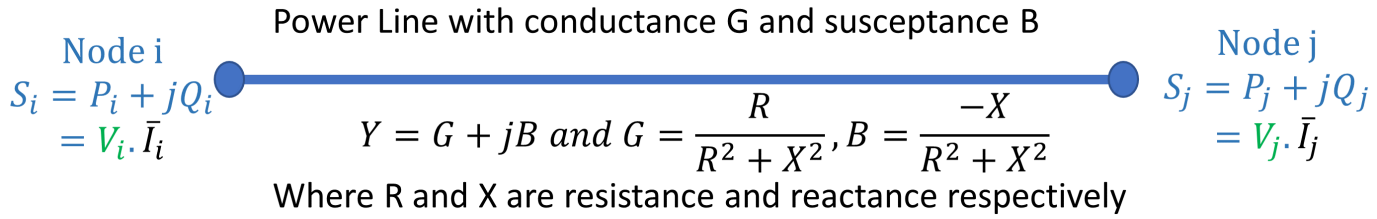
$\mathbf{G}$ and $\mathbf{B}$ conductance and susceptance matrices

$\mathbf{Y} = \mathbf{G} + j\mathbf{B}$

> Coupled quadratics in complex voltage phasors

Power Flow Equations in Different Coordinates

The bus injection model



Polar Voltage Coordinates $\quad V_i = |V_i|\angle\theta_i \quad \theta_1 = 0$

$$P_i = |V_i| \sum_{j=1}^{N} |V_j|(\boldsymbol{G}_{ij} cos(\theta i - \theta j) + \boldsymbol{B}_{ij} sin(\theta i - \theta j))$$
$$Q_i = |V_i| \sum_{j=1}^{N} |V_j|(\boldsymbol{G}_{ij} sin(\theta i - \theta j) - \boldsymbol{B}_{ij} cos(\theta i - \theta j))$$

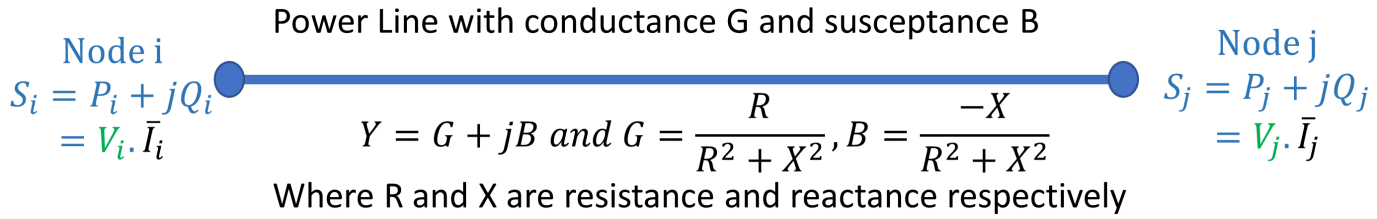
Rectangular Voltage Coordinates $\quad V_i = V_{di} + jV_{qi} \quad V_{q1} = 0$

$$P_i = V_{di}(\boldsymbol{G}_{ij} V_{dj} - \boldsymbol{B}_{ij} V_{qj}) + V_{qi}(\boldsymbol{B}_{ij} V_{dj} + \boldsymbol{G}_{ij} V_{qj})$$
$$Q_i = -V_{di}(\boldsymbol{B}_{ij} V_{dj} + \boldsymbol{G}_{ij} V_{qj}) + V_{qi}(\boldsymbol{G}_{ij} V_{dj} - \boldsymbol{B}_{ij} V_{qj})$$

$V_{di} = Re(V_i) \quad V_{qi} = Im(V_i) \qquad N =$ number of Nodes

Power Flow Equations in Different Coordinates

The bus injection model



Polar Voltage Coordinates	$V_i = \lvert V_i \rvert \angle\theta_i \quad \theta_1 = 0$
$P_i = \lvert V_i \rvert \sum_{j=1}^{N} \lvert V_j \rvert (\boldsymbol{G}_{ij} cos(\theta i - \theta j) + \boldsymbol{B}_{ij} sin(\theta i - \theta j))$	
$Q_i = \lvert V_i \rvert \sum_{j=1}^{N} \lvert V_j \rvert (\boldsymbol{G}_{ij} sin(\theta i - \theta j) - \boldsymbol{B}_{ij} cos(\theta i - \theta j))$	

Rectangular Voltage Coordinates	$V_i = V_{di} + jV_{qi} \quad V_{q1} = 0$
$P_i = V_{di}(\boldsymbol{G}_{ij} V_{dj} - \boldsymbol{B}_{ij} V_{qj}) + V_{qi}(\boldsymbol{B}_{ij} V_{dj} + \boldsymbol{G}_{ij} V_{qj})$	
$Q_i = -V_{di}(\boldsymbol{B}_{ij} V_{dj} + \boldsymbol{G}_{ij} V_{qj}) + V_{qi}(\boldsymbol{G}_{ij} V_{dj} - \boldsymbol{B}_{ij} V_{qj})$	
$V_{di} = Re(V_i) \quad V_{qi} = Im(V_i)$	$N =$ number of Nodes

Power Flow

PF equations are a set of nonlinear algebraic equations which can be solved with Newton-Raphson, GaussSeidel, Fast-decoupled-load-flow method and etc.

Single period ACOPF - problem formulation

Single Period AC Optimal Power Flow (ACOPF)

We are controlling some variables in OPF to minimise system costs with respect to constraints.

Objective Function	$\min_{\mathbf{x}} f(\mathbf{x})$
Equality Constraints (Balance) Power Flow	$\text{s.t. } \mathbf{g}(\mathbf{x}) = \begin{bmatrix} \widetilde{\mathbf{g}}(\mathbf{x}) \\ \overline{\mathbf{g}}(\mathbf{x}) \end{bmatrix} = 0 \qquad \in \mathbb{R}^{n_{gx} \times 1}$
Inequality constraints (line and transformer) Operational Constraints	$\mathbf{h}(\mathbf{x}) = \begin{bmatrix} \widetilde{\mathbf{h}}(\mathbf{x}) \\ \overline{\mathbf{h}}(\mathbf{x}) \end{bmatrix} \leq 0 \qquad \in \mathbb{R}^{n_{hx} \times 1}$
Vector of Variables	$\mathbf{x} = \begin{bmatrix} \boldsymbol{\Theta} & \boldsymbol{\mathcal{V}} & \boldsymbol{\mathcal{P}}^{\mathrm{g}} & \boldsymbol{\mathcal{Q}}^{\mathrm{g}} \end{bmatrix}^{\top} \in \mathbb{R}^{n_x \times 1}$

Limitation of Power Flow

- PF is only a set of static equations which provides status of a system for time: $t = t_s$
- It does not include generation constraints and operational constraints

Limitation of Power Flow

- PF is only a set of static equations which provides status of a system for time: $t = t_s$
- It does not include generation constraints and operational constraints

Limitation of Single Period AC Optimal Power Flow

- OPF is an optimisation problem which optimises status of a system for a SINGLE time: $t = t_s$.
- Although it includes the operational constraints for single time, it will not include operation of DERs and generators over a time horizon.
- It is not capable of integrating storage devices and EVs.

Limitation of Power Flow

- PF is only a set of static equations which provides status of a system for time: $t = t_s$
- It does not include generation constraints and operational constraints

Limitation of Single Period AC Optimal Power Flow

- OPF is an optimisation problem which optimises status of a system for a SINGLE time: $t = t_s$.
- Although it includes the operational constraints for single time, it will not include operation of DERs and generators over a time horizon.
- It is not capable of integrating storage devices and EVs.

Our suggested approach: multi-period ACOPF

Solves the OPF problem over several time-steps at once useful formulation for systems with Energy Storage and Shiftable Loads (e.g. Evs)

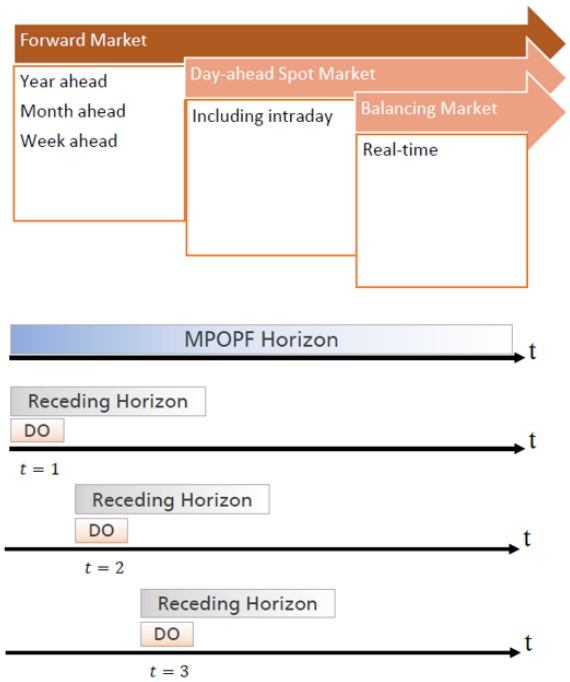
Advantages of MultiPeriod ACOPF

> Integrating dependent time power system's components such as stationary ESS, EV, generators ramp rate, and so on.

Advantages of MultiPeriod ACOPF

> Integrating dependent time power system's components such as stationary ESS, EV, generators ramp rate, and so on.

> between 2% to 5% less costly solutions.



Advantages of MultiPeriod ACOPF

> Integrating dependent time power system's components such as stationary ESS, EV, generators ramp rate, and so on.
> between 2% to 5% less costly solutions.

Disadvantage of MultiPeriod ACOPF

The problem grows very large as the number of time-steps is increased, which may lead to an intractable solution.

MultiPeriod ACOPF-problem formulation

Objective Function	$\min_{\mathbf{X}} F(\mathbf{X})$
Equality Constraints (Balance) Power Flow	$\text{s.t. } G(\mathbf{X}) = \left[\widetilde{G}(\mathbf{X})\ \overline{G}(\mathbf{X})\ \overline{G}^{s}(\mathbf{X})\right]^{\top} = 0 \in \mathbb{R}^{N_g \times 1}$
Inequality constraints (line and transformer) Operational Constraints	$H(\mathbf{X}) = \left[\widetilde{H}(\mathbf{X}) \quad \overline{H}(\mathbf{X})\right]^{\top} \leq 0 \in \mathbb{R}^{N_h \times 1}$
Vectors of Constraints	$\widetilde{\mathbf{G}}(\mathbf{X}) = \left[\widetilde{\mathbf{g}}(\mathbf{x}_1)\ \widetilde{\mathbf{g}}(\mathbf{x}_2)\ \ldots\ \widetilde{\mathbf{g}}(\mathbf{x}_T)\right]^{\top}$ $\overline{\mathbf{G}}(\mathbf{X}) = \left[\overline{\mathbf{g}}(\mathbf{x}_1)\ \overline{\mathbf{g}}(\mathbf{x}_2)\ \ldots\ \overline{\mathbf{g}}(\mathbf{x}_T)\right]^{\top}$ $\overline{\mathbf{G}}^{s}(\mathbf{X}) = \left[\overline{\mathbf{g}}^{s}(\boldsymbol{\tau}_1)\ \overline{\mathbf{g}}^{s}(\boldsymbol{\tau}_2)\ \ldots\ \overline{\mathbf{g}}^{s}(\boldsymbol{\tau}_T)\right]^{\top}$ $\widetilde{\mathbf{H}}(\mathbf{X}) = \left[\widetilde{\mathbf{h}}(\mathbf{x}_1)\ \widetilde{\mathbf{h}}(\mathbf{x}_2)\ \ldots\ \widetilde{\mathbf{h}}(\mathbf{x}_T)\right]^{\top}$ $\overline{\mathbf{H}}(\mathbf{X}) = \left[\overline{\mathbf{h}}(\mathbf{x}_1)\ \overline{\mathbf{h}}(\mathbf{x}_2)\ \ldots\ \overline{\mathbf{h}}(\mathbf{x}_T)\right]^{\top}$

Phase III:

Solution Methods

Nonlinear Programming (NLP)

How do we solve Optimal Power Flow?

I. Interior Point Method

Nonlinear Programming (NLP)

How do we solve Optimal Power Flow?

I. Interior Point Method

II. Gradient Decent Method

Nonlinear Programming (NLP)

How do we solve Optimal Power Flow?

I. Interior Point Method

II. Gradient Decent Method

III. Heuristic Methods

Nonlinear Programming (NLP)

How do we solve Optimal Power Flow?

I. Interior Point Method

II. Gradient Decent Method

III. Heuristic Methods

Interior Point Method

The focus of this study is the interior point method solution

Step(1): Applying Slack variables and the barrier term:

$$\min_{\mathbf{X}} \left[F(\mathbf{X}) - \gamma \sum_{i=1}^{N_h} ln(z_i) \right]$$

$$\text{s.t. } \mathbf{G}(\mathbf{X}) = 0$$

$$\mathbf{H}(\mathbf{X}) + \mathbf{Z} = 0$$

$$\mathbf{Z} \geq 0$$

slack variables "Z" convert inequality constraints to equality constraints.

Step (2): Calculate and form the Lagrangian of Barrier subproblem

$$\mathcal{L}^{\gamma}(\mathbf{X}, \mathbf{Z}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = f(\mathbf{X}) + \boldsymbol{\lambda}^{\top}\mathbf{G}(\mathbf{X}) + \boldsymbol{\mu}^{\top}(\mathbf{H}(\mathbf{X}) + \mathbf{Z}) - \gamma \sum_{i=1}^{N_g} ln(z_i)$$

Step (3): Calculate the KKT[1] of the Lagrangian

Step (3) KKT_X

$$\mathcal{L}^{\gamma}_{\mathbf{X}}(\mathbf{X}, \mathbf{Z}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = f_{\mathbf{X}} + \boldsymbol{\lambda}^{\top}\mathbf{G}_{\mathbf{X}} + \boldsymbol{\mu}^{\top}\mathbf{H}_{\mathbf{X}} = 0$$

Step (3) KKT_Z

$$\mathcal{L}^{\gamma}_{\mathbf{Z}}(\mathbf{X}, \mathbf{Z}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \boldsymbol{\mu}^{\top} - \gamma \mathbf{e}^{\top}\mathbf{diag}(\mathbf{Z})^{-1} = 0$$

Step (3) KKT_λ

$$\mathcal{L}^{\gamma}_{\boldsymbol{\lambda}}(\mathbf{X}, \mathbf{Z}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \mathbf{G}^{\top}(\mathbf{X}) = 0$$

Step (3) KKT_μ

$$\mathcal{L}^{\gamma}_{\boldsymbol{\mu}}(\mathbf{X}, \mathbf{Z}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \mathbf{H}^{\top}(\mathbf{X}) + \mathbf{Z}^{\top} = 0$$

[1] KarushKuhnTucker conditions

Nonlinear Algebric Equations

$$\boldsymbol{\Omega}(\mathbf{X}, \mathbf{Z}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \begin{bmatrix} f_{\mathbf{X}} + \boldsymbol{\lambda}^{\top}\mathbf{G}_{\mathbf{X}} + \boldsymbol{\mu}^{\top}\mathbf{H}_{\mathbf{X}} \\ \mathbf{diag}(\mathbf{Z})\boldsymbol{\mu}^{\top} - \gamma\mathbf{e}^{\top} \\ \mathbf{G}^{\top}(\mathbf{X}) \\ \mathbf{H}^{\top}(\mathbf{X}) + \mathbf{Z}^{\top} \end{bmatrix} = 0$$

$S.t.$

$$\mathbf{Z} > 0$$

$$\boldsymbol{\mu} > 0$$

Step (4): Apply Newton Raphson Method

$$[\boldsymbol{\Omega}_{\mathbf{X}}\ \boldsymbol{\Omega}_{\mathbf{Z}}\ \boldsymbol{\Omega}_{\boldsymbol{\lambda}}\ \boldsymbol{\Omega}_{\boldsymbol{\mu}}]^{k}[\Delta\mathbf{X}\ \Delta\mathbf{Z}\ \Delta\boldsymbol{\lambda}\ \Delta\boldsymbol{\mu}]^{\top k} = -\boldsymbol{\Omega}(\mathbf{X}, \mathbf{Z}, \boldsymbol{\lambda}, \boldsymbol{\mu})^{k}$$

Step (5): Inverse Jacobian of Newton Raphson	$$\begin{bmatrix} \mathbf{M} & \mathbf{G}_{\mathbf{X}}^{\top} \\ \mathbf{G}_{\mathbf{X}} & 0 \end{bmatrix}^{k} \begin{bmatrix} \Delta \mathbf{X} \\ \Delta \boldsymbol{\lambda} \end{bmatrix}^{k} = \begin{bmatrix} -\mathbf{N} \\ -\mathbf{G}(\mathbf{X}) \end{bmatrix}^{k}$$

$\mathbf{M} \in \mathbb{R}^{N_x \times N_x}$ and $\mathbf{N} \in \mathbb{R}^{N_x \times 1}$ are defined as:

$$\mathbf{M} = \boldsymbol{\mathcal{L}}_{\mathbf{XX}}^{\gamma} + \mathbf{H}_{\mathbf{X}}^{\top} \mathbf{diag}(\mathbf{Z})^{-1} \mathbf{diag}(\boldsymbol{\mu}) \mathbf{H}_{\mathbf{X}}$$
$$\mathbf{N} = f_{\mathbf{X}}^{\top} + \mathbf{G}_{\mathbf{X}}^{\top} \boldsymbol{\lambda} + \mathbf{H}_{\mathbf{X}}^{\top} \boldsymbol{\mu} + \mathbf{H}_{\mathbf{X}}^{\top} \mathbf{diag}(\mathbf{Z})^{-1} (\gamma \mathbf{e} + \mathbf{diag}(\boldsymbol{\mu}) \mathbf{H}(\mathbf{X}))$$
$$\boldsymbol{\mathcal{L}}_{\mathbf{XX}}^{\gamma} = f_{\mathbf{XX}} + \mathbf{G}_{\mathbf{XX}}(\boldsymbol{\lambda}) + \mathbf{H}_{\mathbf{XX}}(\boldsymbol{\mu})$$

Iterations

Successive iterations in Interior Point Method

i. **Function Evaluations** calculation of $\mathbf{G_X} = \frac{\partial \mathbf{G}}{\partial \mathbf{X}}$, $\mathbf{H_X} = \frac{\partial \mathbf{H}}{\partial \mathbf{X}}$, $F_{\mathbf{X}} = \frac{\partial F}{\partial \mathbf{X}}$, $\mathbf{G_{XX}} = \frac{\partial}{\partial \mathbf{X}}(\mathbf{G_X^\top} \boldsymbol{\lambda})$, $\mathbf{H_{XX}} = \frac{\partial}{\partial \mathbf{X}}(\mathbf{H_X^\top} \boldsymbol{\lambda})$, $F_{\mathbf{XX}} = \frac{\partial}{\partial \mathbf{X}}(F_{\mathbf{X}}^\top)$ in order to form coefficient matrix and right hand side of $\begin{bmatrix} \mathbf{M} & \mathbf{G_X^\top} \\ \mathbf{G_X} & 0 \end{bmatrix}^k \begin{bmatrix} \Delta \mathbf{X} \\ \Delta \boldsymbol{\lambda} \end{bmatrix}^k = \begin{bmatrix} -\mathbf{N} \\ -\mathbf{G}(\mathbf{X}) \end{bmatrix}^k$

Iterations

Successive iterations in Interior Point Method

i. **Function Evaluations**

ii. **Linear Algebraic Solver** Calculate the Inverse $\begin{bmatrix} \mathbf{M} & \mathbf{G}_{\mathbf{X}}^{\top} \\ \mathbf{G}_{\mathbf{X}} & 0 \end{bmatrix}^{k}$

in $\begin{bmatrix} \mathbf{M} & \mathbf{G}_{\mathbf{X}}^{\top} \\ \mathbf{G}_{\mathbf{X}} & 0 \end{bmatrix}^{k} \begin{bmatrix} \Delta \mathbf{X} \\ \Delta \boldsymbol{\lambda} \end{bmatrix}^{k} = \begin{bmatrix} -\mathbf{N} \\ -\mathbf{G}(\mathbf{X}) \end{bmatrix}^{k}$

Iterations

Successive iterations in Interior Point Method

i. **Function Evaluations**

ii. **Linear Algebraic Solver**

iii. **Miscellaneous** Computational time for other components of IP such as step control and step update: $X^{k+1} = X^k + \Delta X$

Iterations

Successive iterations in Interior Point Method

i. **Function Evaluations**

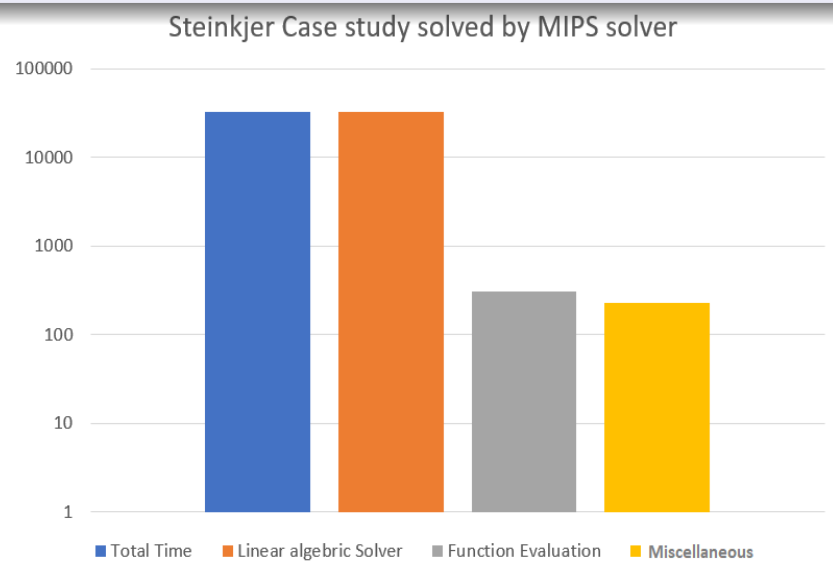
ii. **Linear Algebraic Solver**

iii. **Miscellaneous**

iv. Bottleneck of IP:

- BUS 974
- Branch 1023
- GEN HYDRO 1
- PCC (66kV feeder) 1
- BATTERY 200

Varibles for N= 720 1837440



Phase IV:
Speed-up

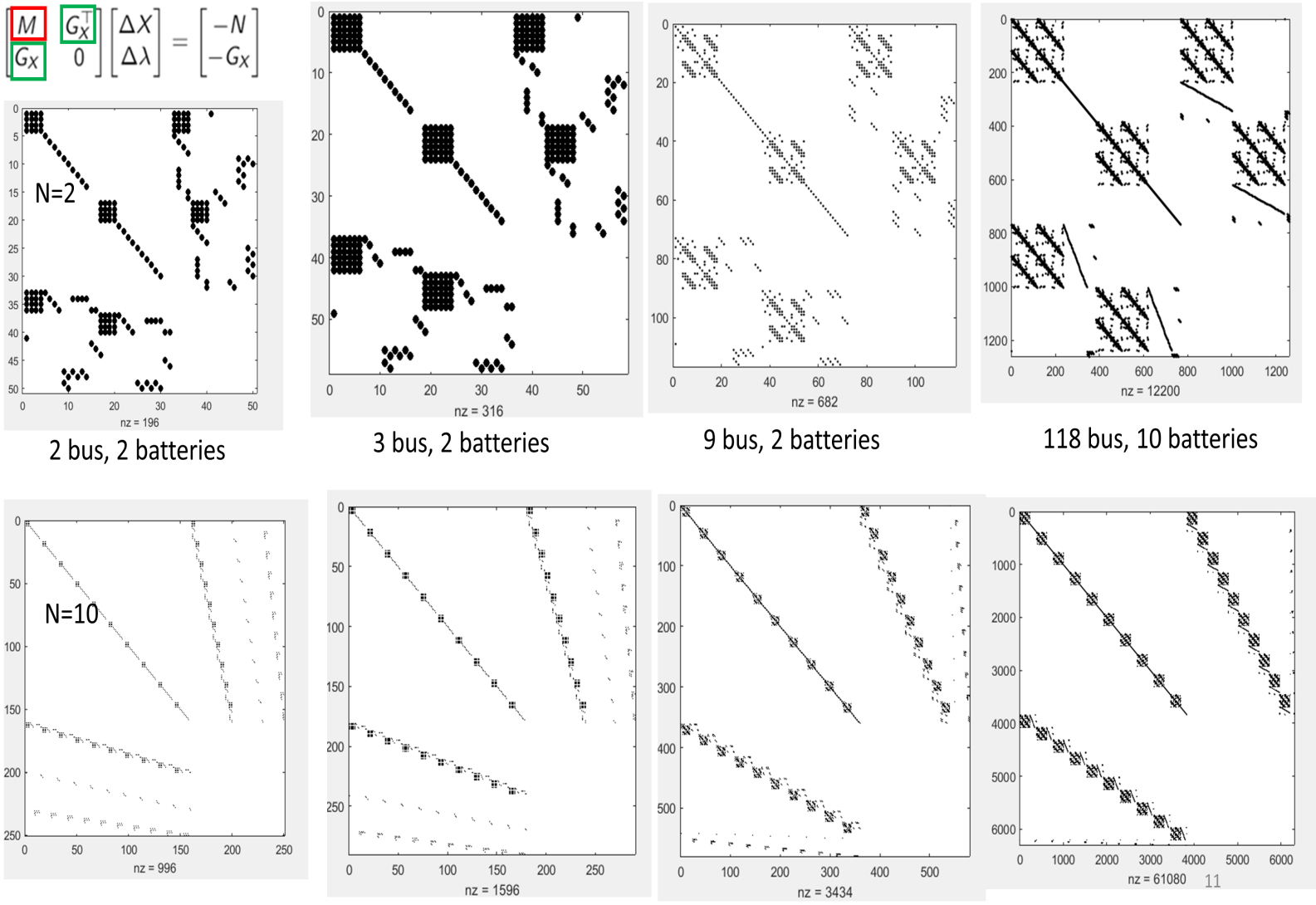


Figure 8: Sparse stracture of the Newton-Raphson Jacobian

Connectivity Matrices

a term in pf equation: $\left[\mathbf{C}_g\right]_{n_b \times n_g} \left[\mathbf{P}_g\right]_{n_g \times 1}$

example:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}_{n_b \times n_g} \begin{bmatrix} \mathbf{P}_{g_1} \\ \mathbf{P}_{g_2} \\ \mathbf{P}_{g_3} \\ \mathbf{P}_{g_4} \\ \mathbf{P}_{g_5} \end{bmatrix}_{n_g \times 1} = \begin{bmatrix} 0 \\ \mathbf{P}_{g_2} \\ 0 \\ \mathbf{P}_{g_1} \\ 0 \\ \mathbf{P}_{g_3} \\ 0 \\ 0 \\ \mathbf{P}_{g_4} \\ \mathbf{P}_{g_5} \end{bmatrix}_{n_b \times 1}$$

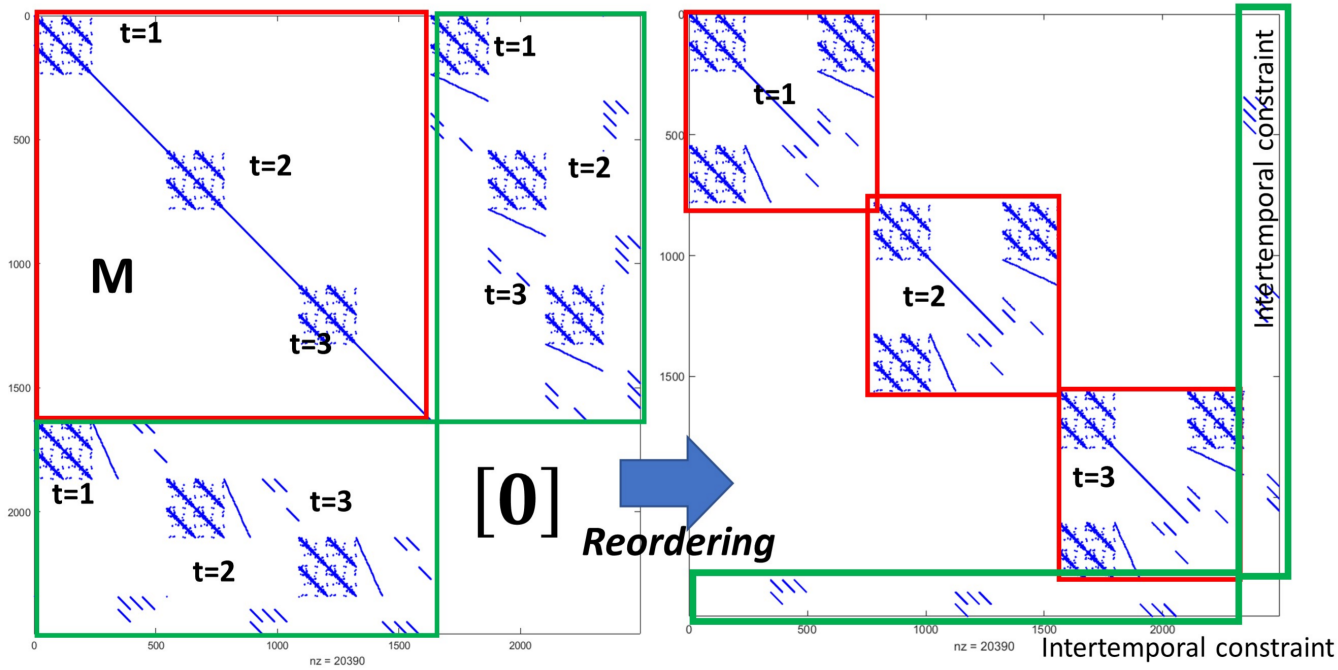


Figure 9: Structure of Jacobian of the Newton-Raphson's algorithm before and after reordering.

Jacobian of Newton-Raphson

$$\begin{bmatrix} \mathbf{M} & \mathbf{G}_{\mathbf{X}}^{\top} \\ \mathbf{G}_{\mathbf{X}} & 0 \end{bmatrix} \begin{bmatrix} \Delta \mathbf{X} \\ \Delta \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} -\mathbf{N} \\ -\mathbf{G}(\mathbf{X}) \end{bmatrix}$$

The solution is published in Papers I and II of this thesis.

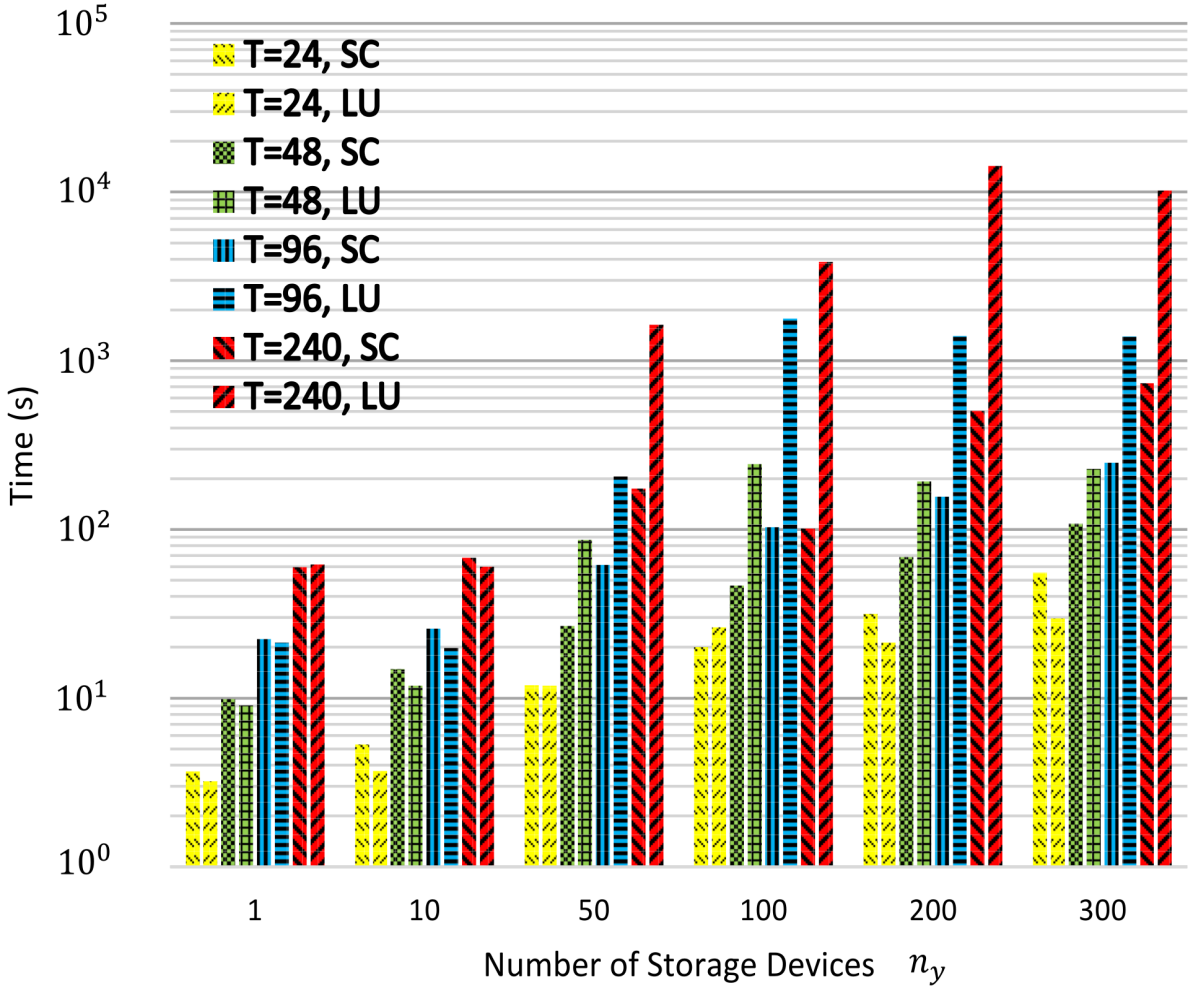


Figure 10: Total time ($\mathrm{TotalTime} = \mathrm{No.of\ Iter.} \times \mathrm{TimePerIter}$) for solution of the linear KKT systems, Case: IEEE 118.

Analytical Derivatives

Table 1: Total time ($\mathrm{TotalTime} = \mathrm{No.of\ Iter.} \times \mathrm{TimePerIter}$) elapsed to calculate: 1) Analytical (hand-coded) derivatives, and 2) Numerical derivatives

				Analytical			Numerical		
Case	T	n_y	iter	$F_{\mathbf{X}}$(s)	$\mathbf{G_X} + \mathbf{H_X}$(s)	$\mathcal{L}^{\gamma}_{\mathbf{XX}}$(s)	$F_{\mathbf{X}}$(s)	$\mathbf{G_X} + \mathbf{H_X}$(s)	$\mathcal{L}^{\gamma}_{\mathbf{XX}}$(s)
Case9	2	5	13	0.03	0.13	0.14	0.43	0.98	140.07
Case9	10	5	23	0.08	0.36	0.37	11.32	30.62	22815.29
IEEE30	2	5	12	0.04	0.25	0.18	1.01	2.16	682.70
IEEE30	10	5	16	0.05	0.24	0.25	16.73	49.12	79712.78
IEEE118	2	5	22	0.04	0.19	0.20	7.41	18.07	24557.09
IEEE118	10	5	37	0.09	0.62	0.82	158.21[1]	572.09[1]	4599735[1]
Pegase 1354	2	5	23	0.05	0.61	0.78	85.54[1]	496.18[1]	7185888[1]
Pegase 1354	10	5	33	0.10	3.77	5.15	588.06[1]	3530[1]	51550941[1]

[1] Estimated total time: The time elapsed for one iteration multiplied to the iteration that would take to converge

Test Cases

Test cases of Case9, IEEE30, IEEE118, and PEGASE1354 are part of open source MATPOWER library.

Phase V:

Future Works

Future work

Goals

i. Translate my entire setup to C and C++.

> Input reading

> Sparsity libraries made particularly for ACOPF and MPOPF mathematical calculations.

> Functions to calculate the first and second order gradients.

> Rebuilt a linear customized solver for this purpose.

Future work

Goals

i. Translate my entire setup to C and C++.
   > Input reading
   > Sparsity libraries made particularly for ACOPF and MPOPF mathematical calculations.
   > Functions to calculate the first and second order gradients.
   > Rebuilt a linear customized solver for this purpose.

ii. Benchmark the entire setup with state of the art Nonlinear Solvers.

Thank you for your attention!

NTNU | Norwegian University of Science and Technology