Power Flow Analysis – Part 2

ELEN90060

Keith Moffat

Class Exercise 1

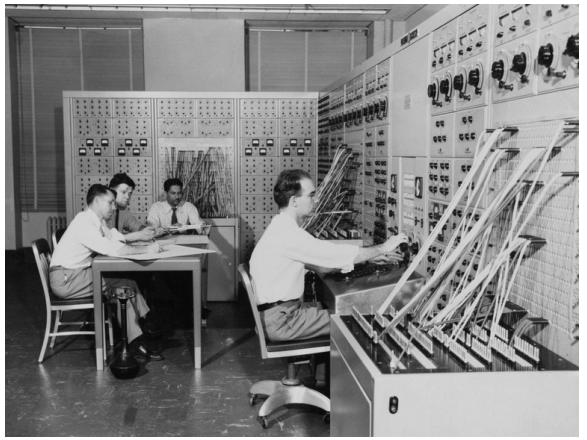
The year is 1949. You work for the **State Electricity Commission of Victoria**, which is rapidly expanding the Victorian electricity grid. New generators are being built in the Latrobe Valley and the Kiewa hydroelectric center, and a future connection to New South Wales is being considered.

You are on the committee planning the transmission network connecting Victoria and New South Wales.

Your boss asks: **Will the proposed transmission system be able to transfer the required power while keeping line currents and customer voltages within acceptable limits?**

“Digital computers” capable of solving this problem are not yet available.

Discuss with your classmates and develop a plan for answering the question.



Photograph Copyright Museums Victoria

History of Power Flow 1/3

Before Digital Computers: Physical Power Flow

- 1920s–1950s: engineers studied operating points using hand calculations and network analyzers.
- Network analyzers were physical scale models of the grid: miniature impedances, sources, and loads.

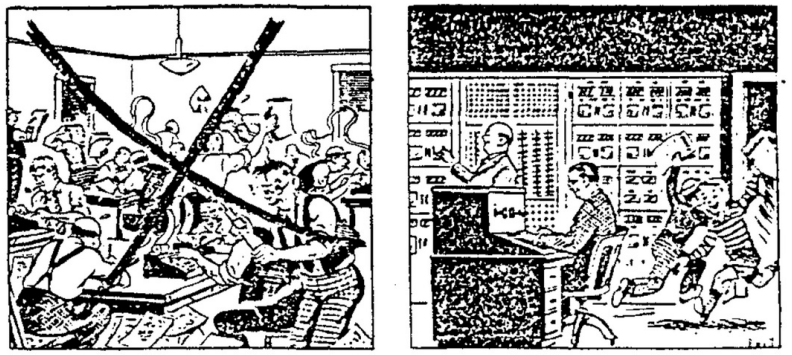


Fig. 1. Drawings of the calculating workshop (left) before and (right) after the introduction of the electrical analyzer.

Aristotle Tympas: From Digital to Analog https://ieeexplore.ieee.org/document/539915

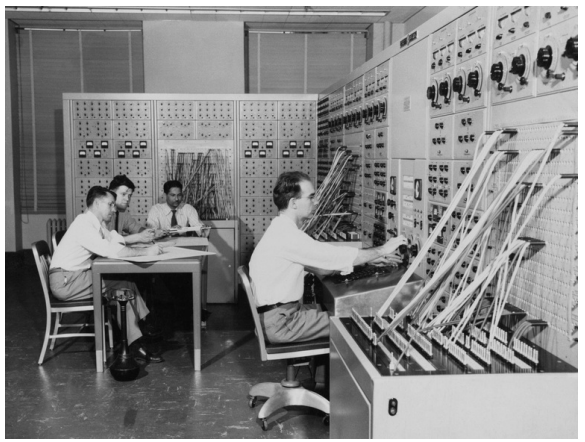
A New Short-circuit Calculating Table

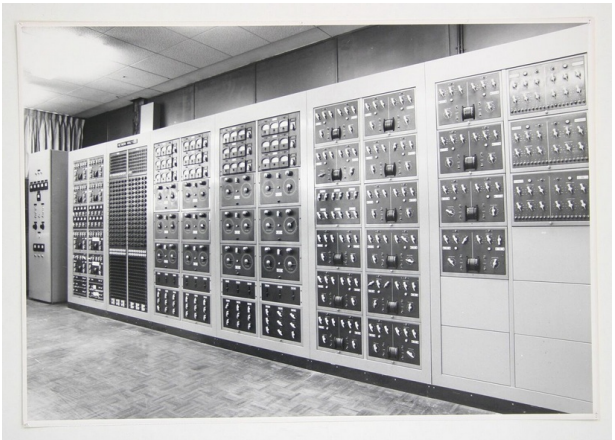
By W. W. Lewis

Power and Mining Engineering Department, General Electric Company



W. W. Lewis, "A new short-circuit calculating table," *General Electric Review*, pp. 669–671, 1920
https://babel.hathitrust.org/cgi/pt?id=nyp.33433062631878&seq=125





Photograph - The Westinghouse AC Network Analyser. Copyright Museums Victoria

History of Power Flow 2/3

1950-1960s: Power Flow Moves to Digital Computers

1956: Ward and Hale publish the first successful digital-computer solution of AC power-flow using Gauss-Seidel

- Run on early drum-memory, vacuum-tube computers
- small networks (~50 nodes)

1967: Tinney and Hart: Newton's method + transistor computers + sparse matrix techniques

- Run on early transistor computers
- large (1000-node) AC power flow



Calculating the Grid: A Late 20th Century Story, Julie Cohn
https://ieeexplore.ieee.org/document/11475571

IEEE TRANSACTIONS ON POWER APPARATUS AND SYSTEMS VOL. PAS-86, NO. 11 NOVEMBER 1967

Power Flow Solution by Newton's Method

WILLIAM. F. TINNEY, SENIOR MEMBER, IEEE, AND CLIFFORD E. HART, MEMBER, IEEE

Abstract—The ac power flow problem can be solved efficiently by Newton's method. Only five iterations, each equivalent to about seven of the widely used Gauss-Seidel method, are required for an exact solution. Problem dependent memory and time requirements vary approximately in direct proportion to problem size. Problems of 500 to 1000 nodes can be solved on computers with 32K core memory. The method, introduced in 1961, has been made practical by optimally ordered Gaussian elimination and special programming techniques. Equations, programming details, and examples of solutions of large problems are given.



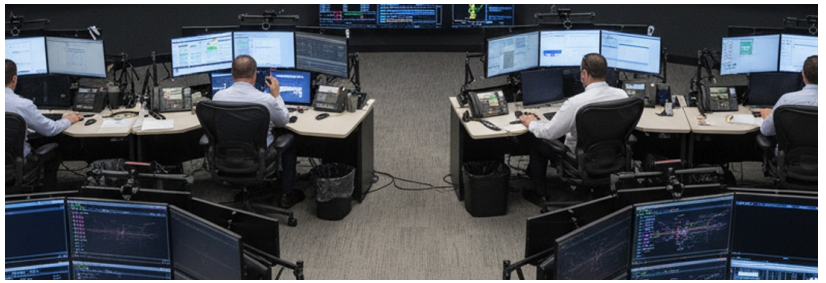
IBM 7040. Photo: Columbia photo archive

History of Power Flow 3/3

Sparse Newton updates are still used ubiquitously:
state estimation, contingency analysis, etc.

1960s: digital power flow





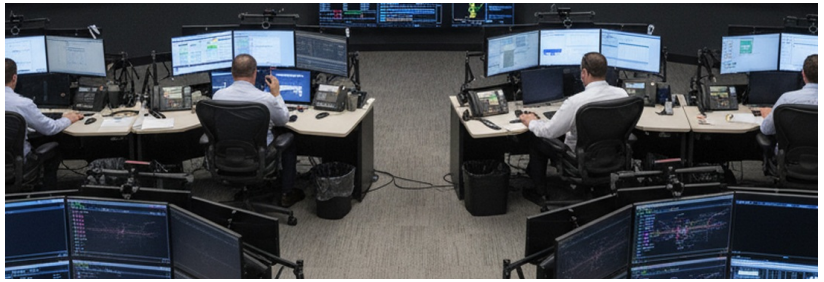
Modern power system operation

Future of Power Flow

Why not just use AI?

1960s: digital power flow





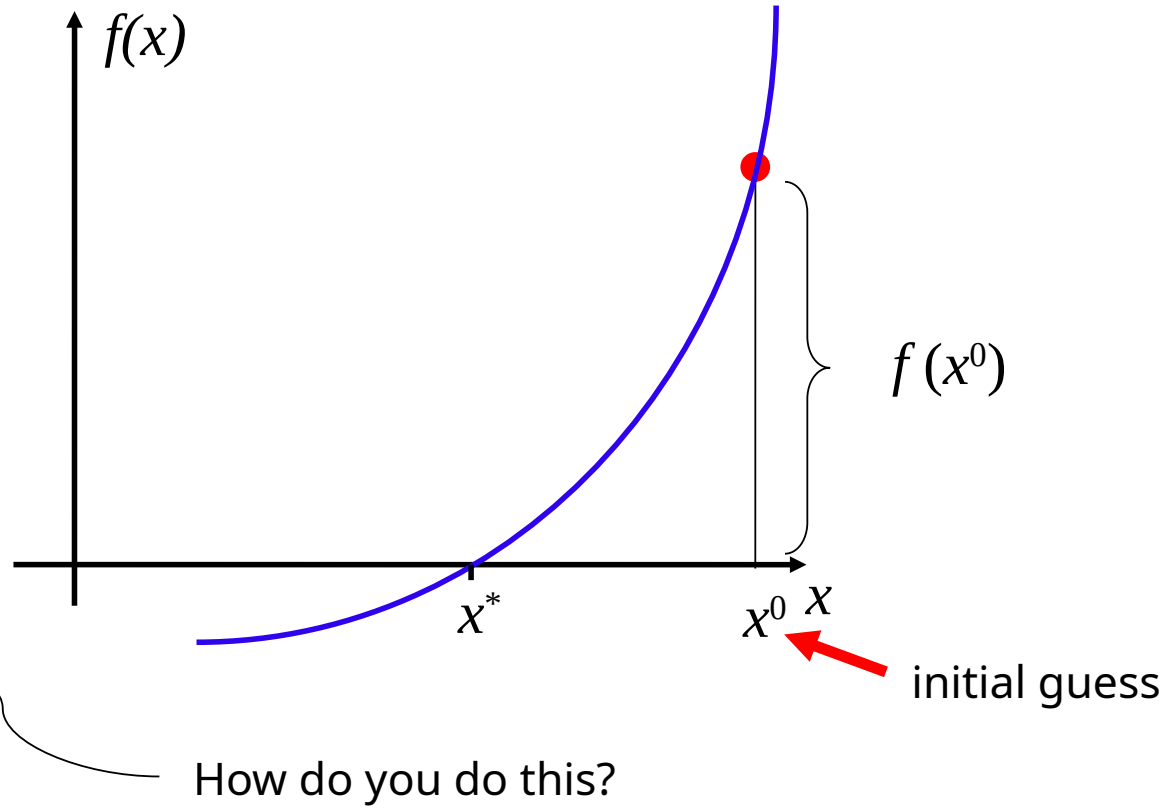
Modern power system operation

?

Newton-Raphson Algorithm 1/2

Context: you

- want to find where $f(x) = 0$
- do not have a plot (e.g., on the right)
- but can query $f(x)$ and its derivative $f'(x)$



Approach:

- Start with an initial guess
- Adjust the guess to decrease the error
- Repeat until the error is small enough

How do you do this?

Newton-Raphson Algorithm 2/2

Initialization: pick an initial guess $x^{(0)}$

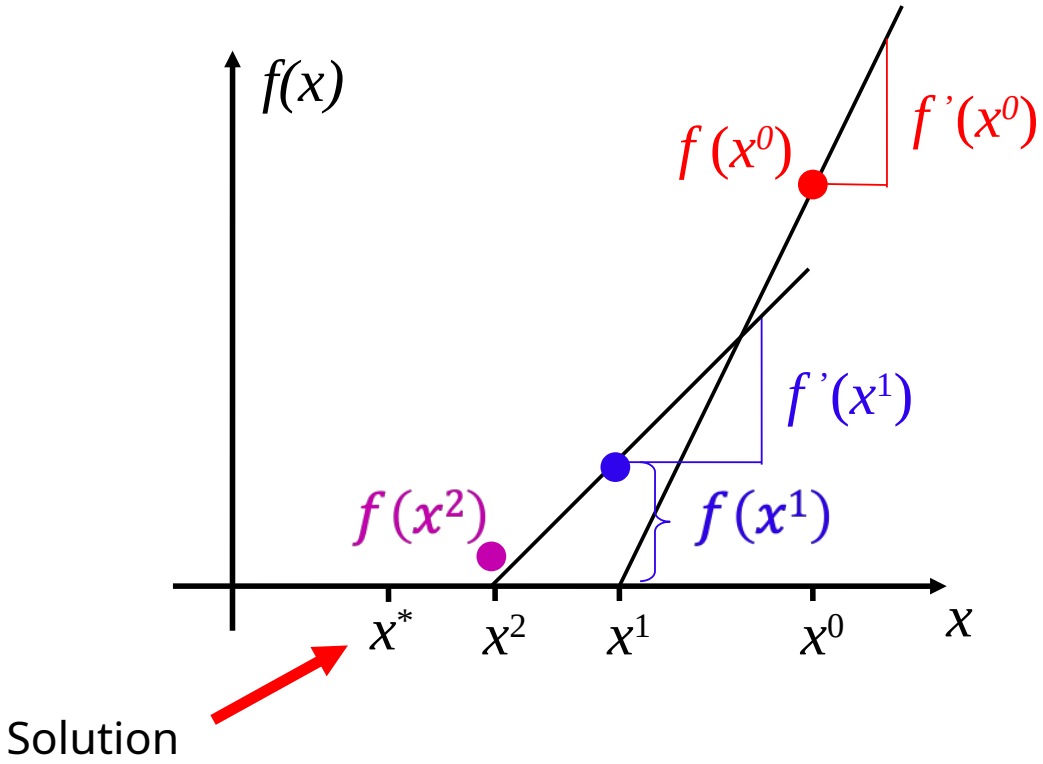
Repeat, until convergence:

1. Solve: $-f'(x^{(k)})\Delta x^{(k)} = f(x^{(k)})$
2. Determine the next guess: $x^{(k+1)} = x^{(k)} + \Delta x^{(k)}$
3. Query the nonlinear function to get $f(x^{(k)})$

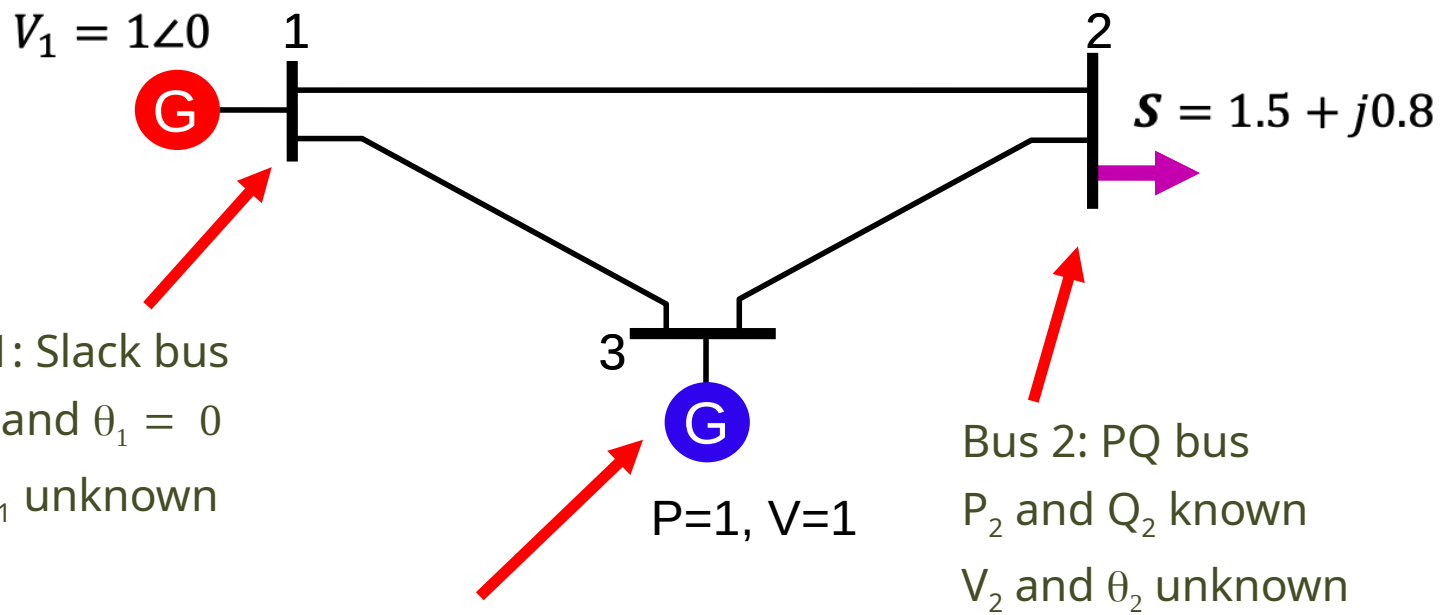
$$-f'\left(x^{(k)}\right)\Delta x^{(k)} = f\left(x^{(k)}\right)$$

$$x^{(k+1)} = x^{(k)} + \Delta x^{(k)}$$

Newton-Raphson Toy Demonstration



Three-Bus Example



Bus 1: Slack bus

V_1=1 and $\theta_1 = 0$

P_1, Q_1 unknown

Bus 2: PQ bus

P_2 and Q_2 known

V_2 and θ_2 unknown

Bus 3: PV bus

P_3 and V_3 known

Q_3 and θ_3 unknown

6 unknowns

Three-Bus Example

The Newton-Raphson iterations are applied to the implicit equations

Implicit equations: known power injections

P at non-slack buses → "P_P"

$$P_2 = 10V_2V_1 \sin\theta_{21} + 4V_2V_3 \sin\theta_{23}$$
$$P_3 = 5V_3V_1 \sin\theta_{31} + 4V_3V_2 \sin\theta_{32}$$

Q at PQ buses → "Q_Q"

$$Q_2 = -10V_2V_1 \cos\theta_{21} + 14V_2^2 - 4V_2V_3 \cos\theta_{23}$$

3 equations
3 unknowns

solve these

then solve these

Explicit equations: unknown power injections

$$P_1 = 10V_1V_2 \sin\theta_{12} + 5V_1V_3 \sin\theta_{13}$$
$$Q_1 = 15V_1^2 - 10V_1V_2 \cos\theta_{12} - 5V_1V_3 \cos\theta_{13}$$
$$Q_3 = -5V_3V_1 \cos\theta_{31} - 4V_3V_2 \cos\theta_{32} + 9V_3^2$$

3 equations
3 unknowns

Implicit Equations → Mismatch Equations

$f_P(\theta, V)$ and $f_Q(\theta, V)$ are the active and reactive power mismatch equations

$$f_P(\theta, V) \equiv P_P(\theta, V) - P_P = 0$$
$$f_Q(\theta, V) \equiv Q_Q(\theta, V) - Q_Q = 0$$

non-slack bus injections calculated from Power Flow Equations

given injections for non-slack buses (constant)

given injections for PQ buses (constant)

PQ bus injections calculated from Power Flow Equations

The power flow solution is the values of V and θ that satisfy the mismatch equations

Goal: find the voltages such that the mismatch equations equal zero

The Power Flow Jacobian Matrix J

For an n-bus system with 1 slack bus, n_{PV} PV buses, and n_{PQ} PQ buses,

- variables: $(n-1)$ unknown voltage angles and n_{PQ} unknown voltage magnitudes, and
- implicit equations: $(n-1)$ for active power and n_{PQ} for reactive power.

Thus, the Power Flow Jacobian has dimension $(n-1+n_{PQ})\times(n-1+n_{PQ})$

denotes matrix dimension

non-slack-bus angles

PQ-bus voltages

$$\frac{\partial \begin{bmatrix} P_P \\ Q_Q \end{bmatrix}}{\partial \begin{bmatrix} \theta_P \\ V_Q \end{bmatrix}} = \begin{bmatrix} \frac{\partial P_P}{\partial \theta_P} & \frac{\partial P_P}{\partial V_Q} \\ \frac{\partial Q_Q}{\partial \theta_P} & \frac{\partial Q_Q}{\partial V_Q} \end{bmatrix} = \begin{bmatrix} J_1\left(\begin{bmatrix} \theta_P \\ V_Q \end{bmatrix}\right) & J_2\left(\begin{bmatrix} \theta_P \\ V_Q \end{bmatrix}\right) \\ J_3\left(\begin{bmatrix} \theta_P \\ V_Q \end{bmatrix}\right) & J_4\left(\begin{bmatrix} \theta_P \\ V_Q \end{bmatrix}\right) \end{bmatrix}$$

$\mathbb{R}^{(n-1)\times(n-1)}$ $\mathbb{R}^{(n-1)\times n_{PQ}}$ $\mathbb{R}^{n_{PQ}\times(n-1)}$ $\mathbb{R}^{n_{PQ}\times n_{PQ}}$

Solving the Jacobian linear system

4. Δx is calculated by solving the linear system:

$$-J(x)\Delta x = f(x)$$

$$-\begin{bmatrix} J_1 & J_2 \\ J_3 & J_4 \end{bmatrix}\begin{bmatrix} \Delta\theta_P \\ \Delta V_Q \end{bmatrix} = \begin{bmatrix} f_P(\theta, V) \\ f_Q(\theta, V) \end{bmatrix}$$

One way to do this is by inverting the Jacobian:

$$\Delta x = -J^{-1}(x)f(x)$$

In practice, there are faster ways of solving linear systems than matrix inversion.

Solving Linear Equations Repeatedly

Changing Matrix $J^{(k)}$

Before the iterations: nothing

At every iteration k:

1. Build $J^{(k)}$
2. Factorize $J^{(k)} = L^{(k)}U^{(k)}$
3. Solve $L^{(k)}U^{(k)}\Delta x^{(k)} = b^{(k)}$

Repeats the expensive factorization every iteration.

Constant Matrix J

Before the iterations:

- Build J
- Factorize $J = LU$ once

At every iteration k:

1. Solve $LU\Delta x^{(k)} = b^{(k)}$

Only factorizes the matrix once.

(why not just invert J?)

Sparsity!

Class Exercise 2

The year is 1973. You work for the **State Electricity Commission of Victoria**. Digital computers and Newton's method can now solve the AC power-flow equations for large networks using Tinney and Hart's sparsity tricks. However, the available computers have limited processing power and memory.

A single power-flow solution is manageable, but system planners must now perform an N−1 security study by repeating the calculation for every credible single transmission-line or generator outage. Using the standard Newton method, completing all these calculations takes too long.

Your boss asks: **How can we accelerate the AC power-flow calculation sufficiently to perform a complete N−1 study using the available computing resources?**

Discuss with your classmates and develop a plan for answering this question.

Hint: matrix pre-factorization



IBM 7040. Photo: Columbia photo archive

Decoupled Power Flow

1972: Stott publishes *Decoupled Newton Load Flow*

1974: Stott and Alsac publish *Fast Decoupled Load Flow*

DECOUPLED NEWTON LOAD FLOW

Brian Stott
Power Systems Laboratory
University of Manchester Institute of Science and Technology, Manchester, U.K.

$$\begin{bmatrix} \Delta P^k \\ \Delta Q^k \end{bmatrix} = \begin{bmatrix} H^k & N^k \\ J^k & L^k \end{bmatrix} \cdot \begin{bmatrix} \Delta \theta^{k+1} \\ \Delta V^{k+1} \end{bmatrix}$$

FAST DECOUPLED LOAD FLOW

B. STOTT O. ALSAÇ

Power Systems Laboratory
University of Manchester Institute of Science and Technology, Manchester, U.K.

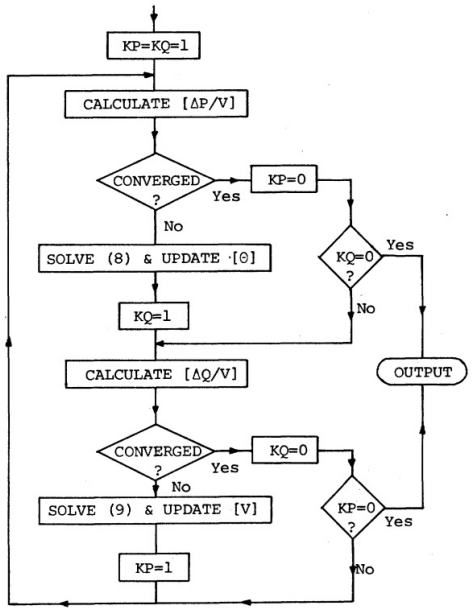


Fig. 1 Flow diagram of the iteration scheme

Decoupled Power Flow

Are these algorithms still relevant?

Yes.

Decoupled and fast-decoupled power-flow methods are still available in widely used power-system software and remain useful, particularly for large transmission networks and repeated studies.

However, modern computing has made their computational advantage less decisive, and full Newton methods are often preferred for their greater robustness.

Outline

1. Decoupled Power flow
2. Fast Decoupled Power Flow
3. DC Power Flow

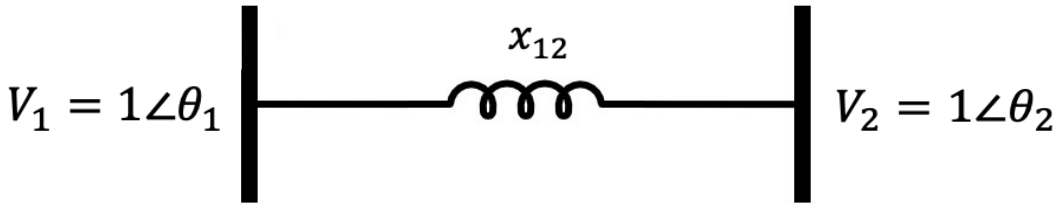
Decoupled Power Flow

Consider the standard AC Power Flow Newton-Raphson iteration $-J(x)\Delta x = f(x)$:

$$-\begin{bmatrix} J_1\left(\theta_P^{(k)}, V_Q^{(k)}\right) & J_2\left(\theta_P^{(k)}, V_Q^{(k)}\right) \\ J_3\left(\theta_P^{(k)}, V_Q^{(k)}\right) & J_4\left(\theta_P^{(k)}, V_Q^{(k)}\right) \end{bmatrix} \begin{bmatrix} \Delta\theta_P^{(k)} \\ \Delta V_Q^{(k)} \end{bmatrix} = \begin{bmatrix} f_P\left(\theta_P^{(k)}, V_Q^{(k)}\right) \\ f_Q\left(\theta_P^{(k)}, V_Q^{(k)}\right) \end{bmatrix}$$

Can we simplify this equation?

Decoupled Power Flow
Two-Bus Transmission System with an inductive line



$$P_k^{injected} = \sum_{i \in N_k} V_k V_i [G_{ki} \cos\theta_{ki} + B_{ki} \sin\theta_{ki}]$$

$$P_{21} = V_2 \cdot V_1 \cdot [G_{21} \cos(\theta_{21}) + B_{21} \sin(\theta_{21})]$$

$$= -B_{21} V_2 V_1 \sin(\theta_2 - \theta_1)$$

$\sin(\theta_2 - \theta_1) \approx \theta_2 - \theta_1$

$$\approx -B_{21}(\theta_2 - \theta_1)$$

⇒ *P*-θ coupling

$$Q_k^{injected} = \sum V_k V_i [G_{ki} \sin\theta_{ki} - B_{ki} \cos\theta_{ki}]$$

$$Q_{21} = V_2 \cdot V_1 \cdot [G_{21} \sin(\theta_{21}) - B_{21} \cos(\theta_{21})]$$

$$= -B_{21}[V_2 V_1 - V_1 \cos(\theta_2 - \theta_1)]$$

$\cos(\theta_2 - \theta_1) \approx 1$

$$\approx -B_{21}[V_2 - V_1]$$

⇒ *Q*-*V* coupling

Power flow equation reminder

Where did these $P_{21} = V_2 V_1[G_{21}\cos(\theta_{21}) + B_{21}\sin(\theta_{21})]$ $Q_{21} = V_2 V_1[G_{21}\sin(\theta_{21}) - B_{21}\cos(\theta_{21})]$ equations come from?

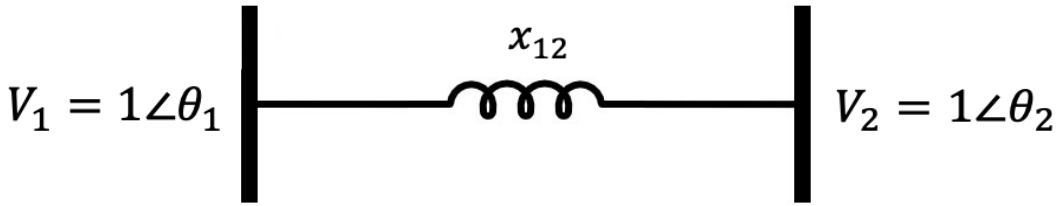
The Bus Injection Equations:

$$P_k^{injected} = \sum_{i \in N_k} V_k V_i [G_{ki}\cos\theta_{ki} + B_{ki}\sin\theta_{ki}]$$

$$Q_k^{injected} = \sum_{i \in N_k} V_k V_i [G_{ki}\sin\theta_{ki} - B_{ki}\cos\theta_{ki}]$$

which come from the complex power equations: $\boldsymbol{S = V\bar{I}}$ $\boldsymbol{I = YV}$

Decoupled Power Flow
Two-Bus Transmission System with an inductive line



The voltage angles approximately determine the real power flow: $P_{21} \approx -B_{21}(\theta_2 - \theta_1)$

The voltage magnitudes approximately determine the reactive power flow: $Q_{21} \approx -B_{21}[V_2 - V_1]$

Decoupled Power Flow

For inductive transmission lines, the Jacobian is approximately decoupled

$$-\begin{bmatrix} J_1 & 0 \\ 0 & J_4 \end{bmatrix}\begin{bmatrix} \Delta\theta_P \\ \Delta V_Q \end{bmatrix} = \begin{bmatrix} f_P(\theta, V) \\ f_Q(\theta, V) \end{bmatrix}$$

$$-J_1\Delta\theta_P = f_P(\theta, V)$$

$$-J_4\Delta V_Q = f_Q(\theta, V)$$

Two smaller systems of linear equations is easier to solve than one large one.

Note: we are only changing the Jacobian used for the Newton step, not the power mismatch equations that we are trying to solve.

Decoupled Power Flow Algorithm

Initialization:

Pick an initial guess $x^{(0)}$. Determine the approximate Jacobian function $\tilde{f}'(x)$: $\begin{bmatrix} J_1 & 0 \\ 0 & J_4 \end{bmatrix}$

Repeat, until convergence:

$$-J_1 \Delta\theta_P = f_P(\theta, V)$$
$$-J_4 \Delta V_Q = f_Q(\theta, V)$$

1. Solve the Newton step using the approximate Jacobian: $-\tilde{f}'(x^{(k)}) \cdot \Delta x^{(k)} = f(x^{(k)})$
2. Determine the next guess: $x^{(k+1)} = x^{(k)} + \Delta x^{(k)}$
3. Query the true power mismatch equations: $f(x^{(k)})$

Are there general solution guarantees?

No.

We expect convergence for fewer initial conditions than full Newton, but often it's good enough.

Decoupled Power Flow
Scalar Analogy

$$f(x) = x^2 - 5x + 4 = 0 \qquad x^{(0)} = 6$$

$$-\tilde{f}'(x^{(k)})\Delta x^{(k)} = f(x^{(k)})$$

$$x^{(k+1)} = x^{(k)} + \Delta x^{(k)}$$

Approx. gradient

True function evaluation

True derivative: $f'(x) = 2x - 5$ Approx. derivative: $\tilde{f}'(x) = 3x - 5$

What happens if we use the approximate derivative?

$$f(x^{(0)}) = 10$$

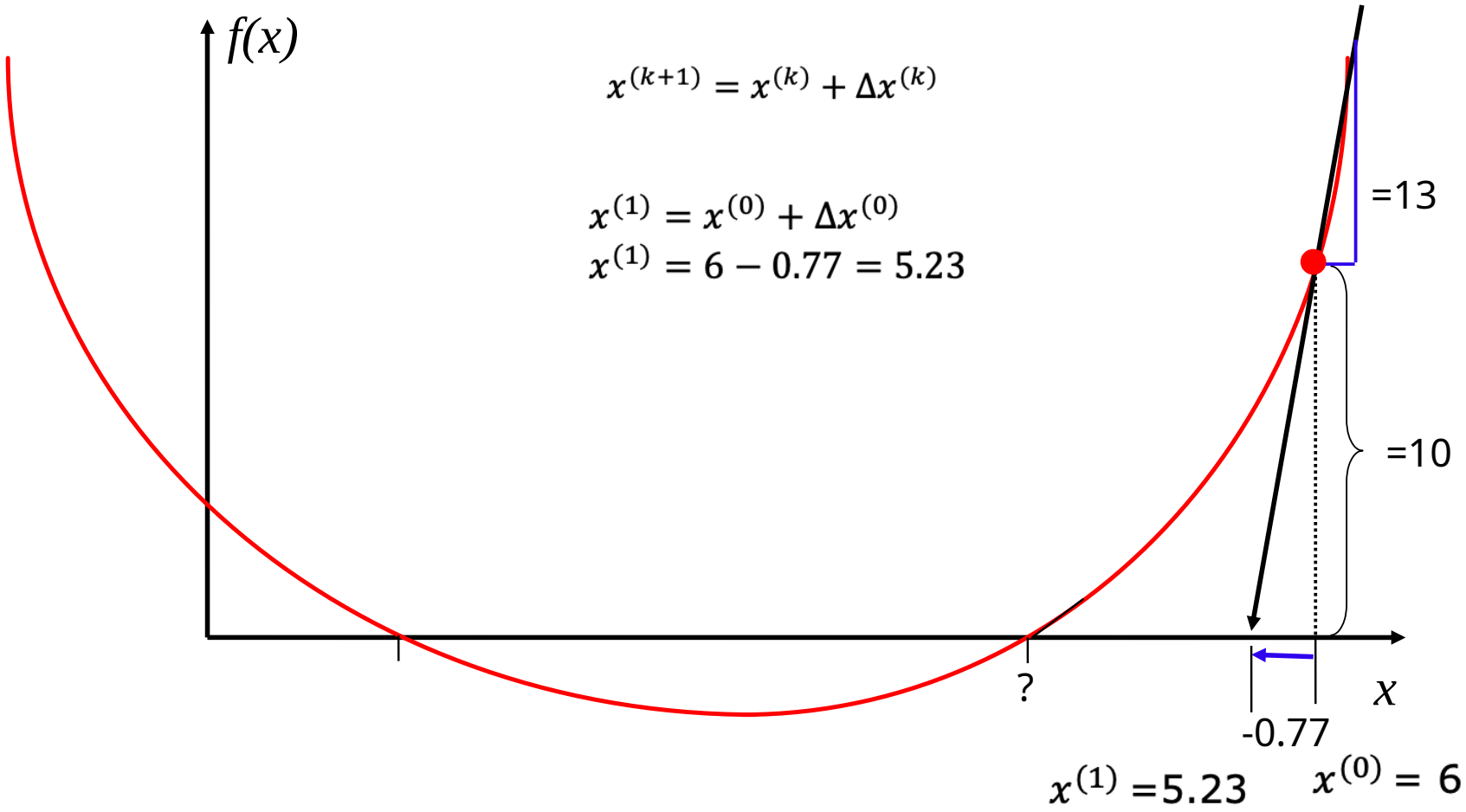
$$f'(x^{(0)}) = 7 \qquad \tilde{f}'(x^{(0)}) = 13$$

$$-7\Delta x^{(0)} = 10 \qquad -13\Delta x^{(0)} = 10$$

$$\Delta x^{(0)} = -10/7 = -1.43 \qquad \Delta x^{(0)} = -10/13 = -0.77$$

The approx. derivative still moves x in correct direction, just not as strongly

Decoupled Power Flow
Scalar Analogy

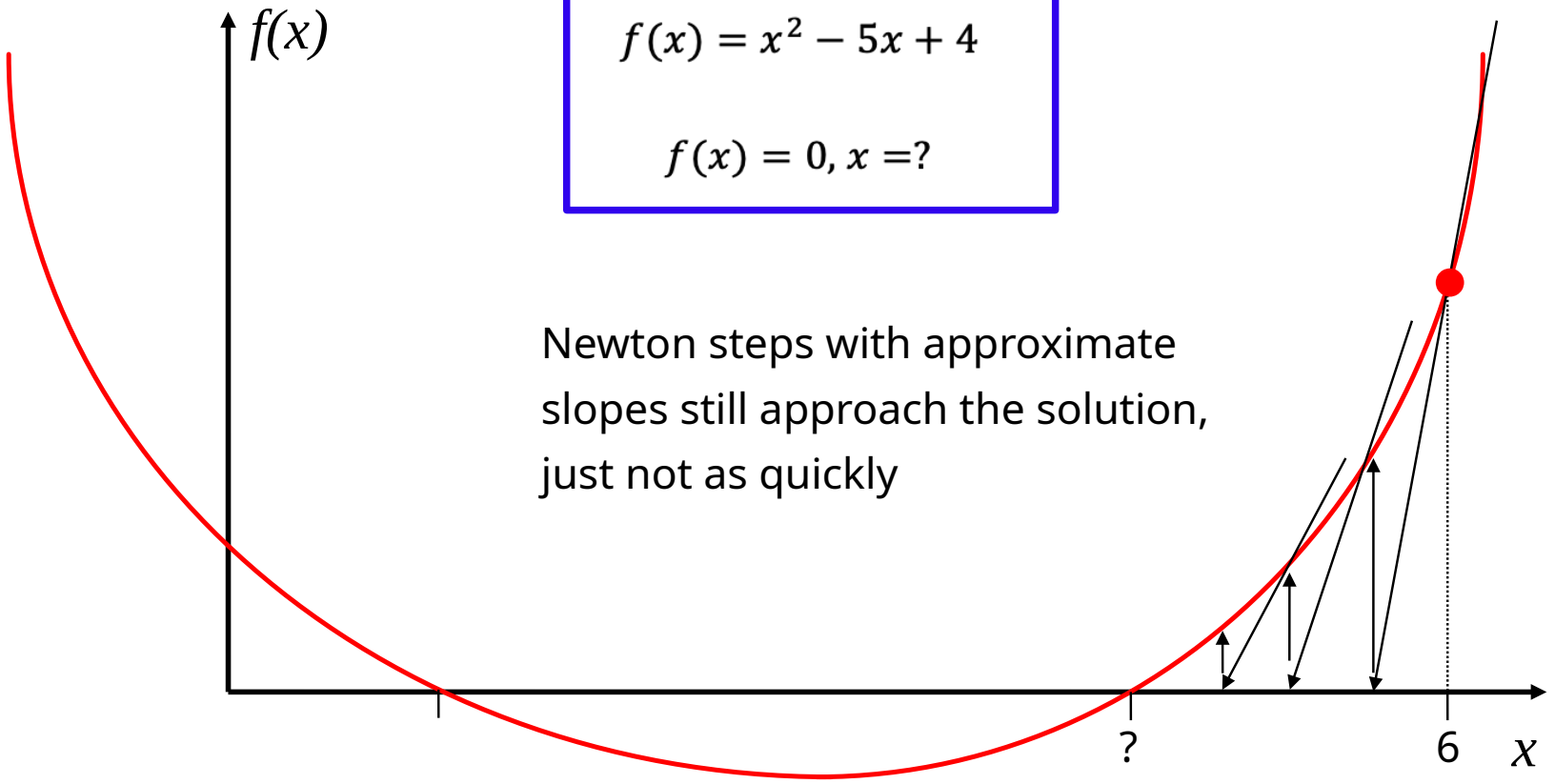


Decoupled Power Flow
Scalar Analogy

$$f(x) = x^2 - 5x + 4$$

$$f(x) = 0, x = ?$$

Newton steps with approximate slopes still approach the solution, just not as quickly



Decoupled Power Flow

Does Decoupled Power Flow still converge to the true solution?

Yes—if it converges.

Decoupled power flow evaluates the true nonlinear power mismatches but uses an approximate Jacobian to calculate each update. Therefore, at convergence, the active- and reactive-power mismatches are zero and the solution satisfies the full AC power-flow equations.

The approximate Jacobian affects convergence speed and robustness, not the final solution.

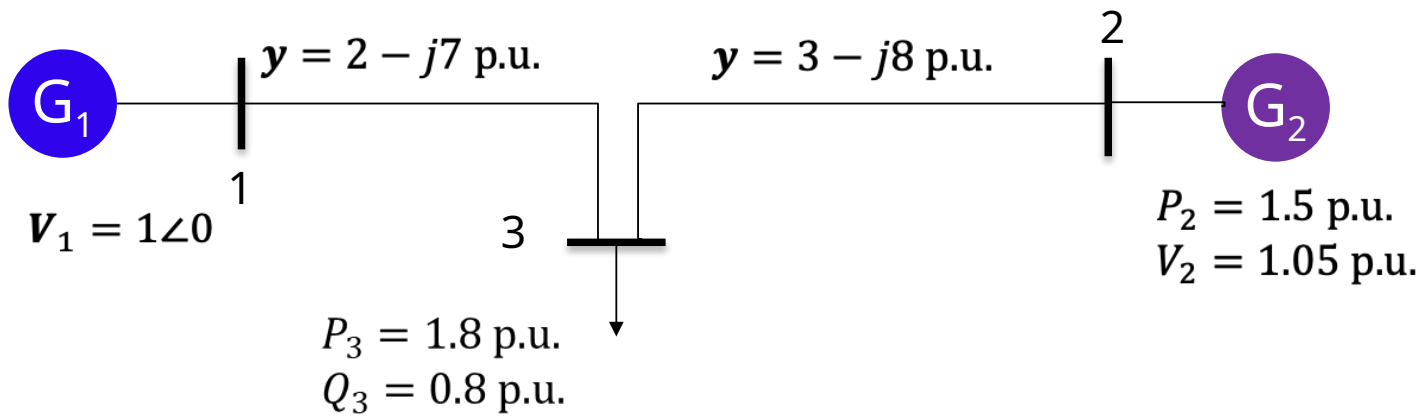
Decoupled Power Flow Example

Consider the single-line diagram shown

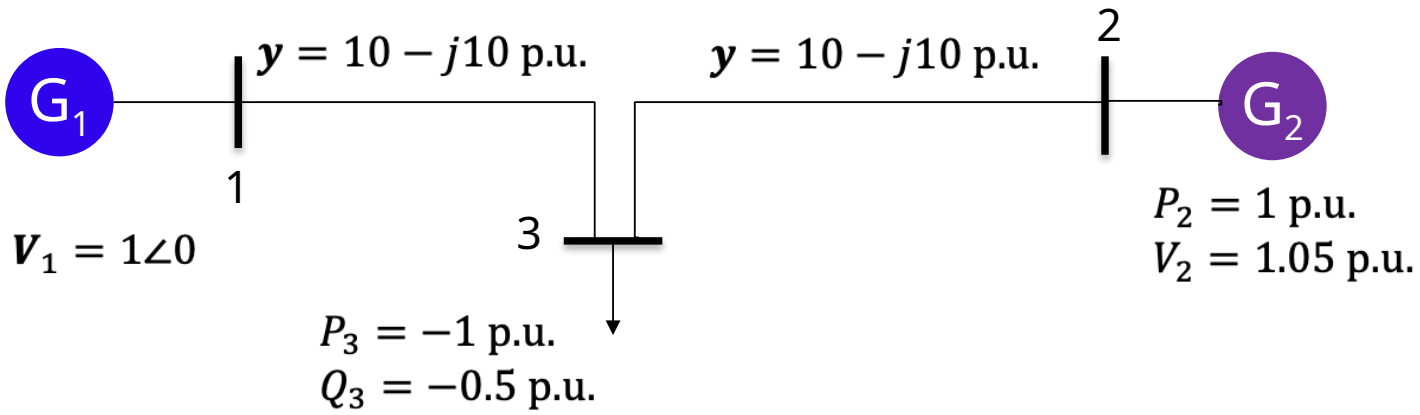
There is a reactive power limit on the generation unit located at bus 2

$$-0.5\ p.u. \leq Q_2 \leq 0.95\ p.u.$$

➢Calculate voltages for the first iteration of Decoupled Power Flow Method.



Decoupled Power Flow Example



Calculate voltages for the first iteration of Decoupled Power Flow Method.

Fast Decoupled Power Flow

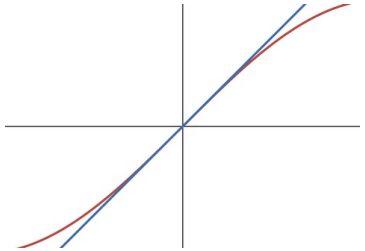
Takes an additional Jacobian simplification step beyond Decoupled Power Flow

J_1 and J_4 depend on V and θ, and thus change every Newton iteration

Can we find a constant approximation of J_1 and J_4?

→ the Jacobian at the nominal voltage point.

Observation: voltage angle differences are often small: $\theta_k - \theta_i \approx 0$



Fast Decoupled Power Flow

Transmission-network approximations:

$$R_{ki} \ll X_{ki} \Rightarrow G_{ki} \approx 0, \qquad \sin(\theta_{ki}) \approx \theta_{ki}, \qquad \cos(\theta_{ki}) \approx 1, \qquad V_i \approx 1\text{p.u.}$$

Calculate the non-zero Jacobian entries from

$$P_k^{injected} = \sum_{i \in N_k} V_k V_i [G_{ki} \cos\theta_{ki} + B_{ki} \sin\theta_{ki}]$$

$$Q_k^{injected} = \sum V_k V_i [G_{ki} \sin\theta_{ki} - B_{ki} \cos\theta_{ki}]$$

Active power / voltage angle

Reactive power / voltage magnitude

for the off-diagonal entries $i \neq k$

$$[J_1]_{ki} = \frac{\partial P_k}{\partial \theta_i} = V_k V_i [G_{ki}\sin(\theta_{ki}) - B_{ki}\cos(\theta_{ki})]$$

$$\approx -V_k V_i B_{ki} \approx -V_k B_{ki}$$

$$[J_4]_{ki} = \frac{\partial Q_k}{\partial V_i} = V_k [G_{ki}\sin(\theta_{ki}) - B_{ki}\cos(\theta_{ki})]$$

$$\approx -V_k B_{ki} \approx -V_k B_{ki}$$

the diagonal entries $i = k$ admit the same approximation (when there are no shunt admittances)

$$[J_1]_{kk} \approx -V_k B_{kk}$$

$$[J_4]_{kk} \approx -V_k B_{kk}$$

Fast Decoupled Power Flow

Transmission-network approximations:

$$R_{ki} \ll X_{ki} \Rightarrow G_{ki} \approx 0, \qquad \sin(\theta_{ki}) \approx 0, \qquad \cos(\theta_{ki}) \approx 1, \qquad V_i \approx 1\text{p.u.}$$

$$[J_1]_{kk} \approx -V_k B_{kk}$$

$$J_1 \approx -\text{diag}(V)B$$

$$[J_4]_{kk} \approx -V_k B_{kk}$$

$$J_4 \approx -diag(V)B$$

the approximate Newton steps

$$-J_1 \Delta\theta_P = f_P(\theta, V)$$

$$-J_4 \Delta V_Q = f_Q(\theta, V)$$

Two options for Fast Decoupled Power Flow

Option 1: approximate $\text{diag}(V) = \text{diag}(1)$

$$B_P \Delta\theta_P = f_P(\theta, V)$$
$$B_q \Delta V_Q = f_Q(\theta, V)$$

(less accurate, simpler)

Option 2: keep $\text{diag}(V) = \text{diag}(1)$

$$B_P \Delta\theta_P = \text{diag}(V_P)^{-1} f_P(\theta, V)$$
$$B_Q \Delta V_Q = \text{diag}(V_Q)^{-1} f_Q(\theta, V)$$

(more accurate)

B_P and B_Q can be pre-factored and thus each Newton step is very fast

Fast Decoupled Power Flow Algorithm

Initialization:

Pick an initial guess $x^{(0)}$. Determine the constant approximate Jacobian function $\tilde{f}'$: $\begin{bmatrix} -B_P & 0 \\ 0 & -B_Q \end{bmatrix}$

Repeat, until convergence:

1. Solve the Newton step using the constant, approximate Jacobian : $-\tilde{f}' \cdot \Delta x^{(k)} = f(x^{(k)})$
2. Determine the next guess: $x^{(k+1)} = x^{(k)} + \Delta x^{(k)}$
3. Query the true power mismatch equations: $f(x^{(k)})$

Are there general solution guarantees?

No.

We expect it convergence for fewer initial conditions than full Newton, but it's often good enough.

Fast Decoupled Power Flow
Scalar Analogy

$$f(x) = x^2 - 5x + 4 = 0 \qquad x^{(0)} = 6$$

$$-\tilde{f}\Delta x^{(k)} = f(x^{(k)})$$

$$x^{(k+1)} = x^{(k)} + \Delta x^{(k)}$$

Constant approx. gradient

True function evaluation

True derivative: $f' = 2x - 5$ Approx. derivative: $\tilde{f}' = 10$

What happens if we use the approximate derivative?

$$f(x^{(0)}) = 10$$

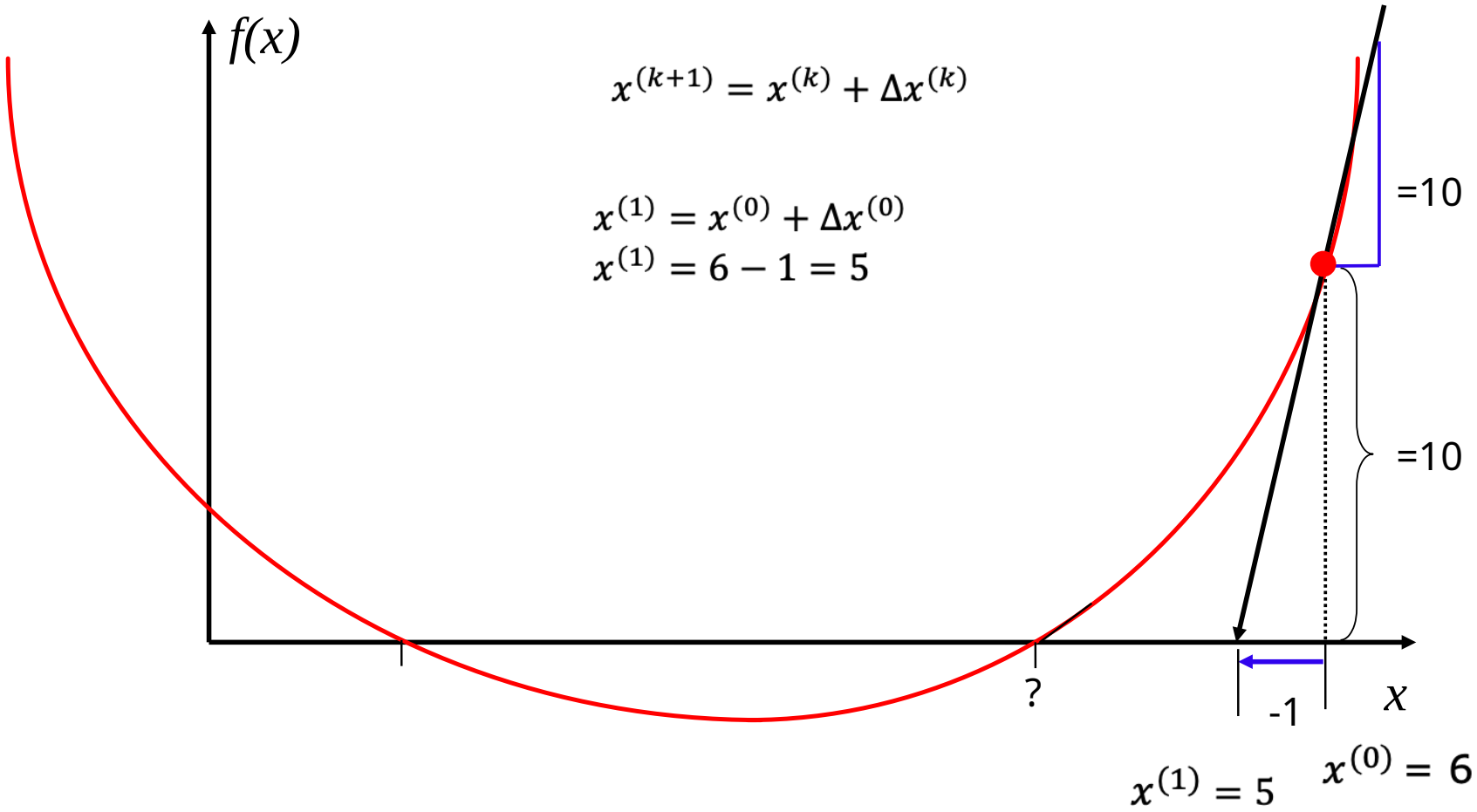
$$f'(x^{(0)}) = 7 \qquad \tilde{f}'(x^{(0)}) = 10$$

$$-7\Delta x^{(0)} = 10 \qquad -10\Delta x^{(0)} = 10$$

$$\Delta x^{(0)} = -1.43 \qquad \Delta x^{(0)} = -1$$

The approx. derivative still moves x in correct direction, just not as strongly

Decoupled Power Flow
Scalar Analogy

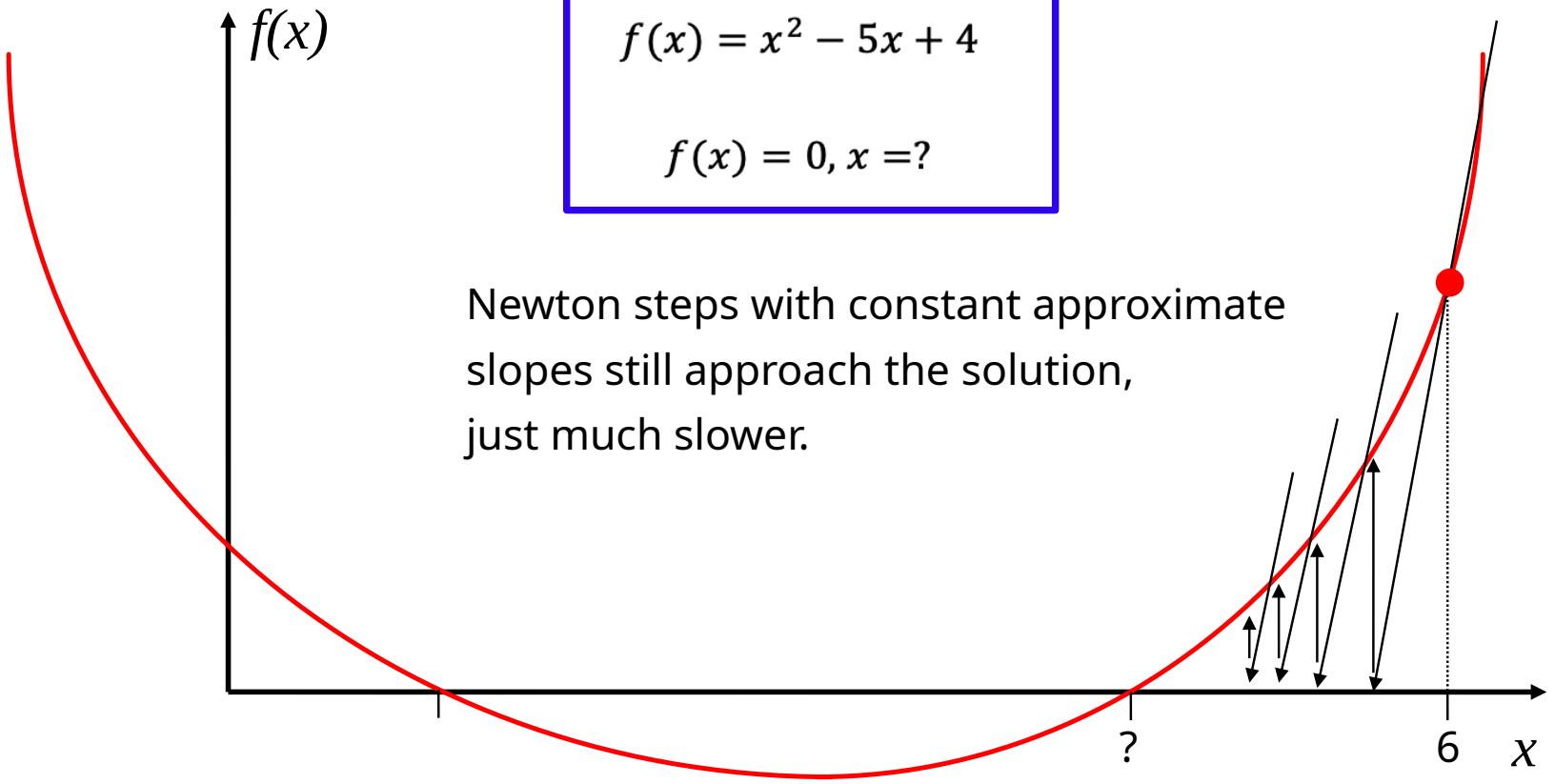


Decoupled Power Flow
Scalar Analogy

$$f(x) = x^2 - 5x + 4$$

$$f(x) = 0, x = ?$$

Newton steps with constant approximate slopes still approach the solution, just much slower.



Fast Decoupled Power Flow

Does Fast Decoupled Power Flow still converge to the true solution?

Yes—if it converges.

Like Decoupled Power Flow, Fast Decoupled Power Flow evaluates and iterates based on the true nonlinear power mismatches.

Unlike Decoupled Power Flow, it uses fixed approximate Jacobian matrices.

The approximate Jacobian affects convergence speed and robustness, not the final solution.

Class Exercise 3



The year is 2040. You work for the **State Electricity Commission of Victoria**. Everyone has forgotten how to do maths and AC Power Flow is too confusing for people now.

Your boss asks: **Hey boss if we build more data centers will the grid melt? Our computer is broken btw.**

Discuss with your classmates and develop a plan for answering this question.

Hint: Can you approximate real-power flow with a linear model?



W. W. Lewis, “A new short-circuit calculating table,” 1920

Extra Credit: What cult classic is the screenshot above from?
Hint: Plants love Brawndo

Power Flow Analysis – Part 2

ELEN90060

Keith Moffat