4.3 Time-Domain Simulation
Contents

• Explicit and implicit numerical integration techniques
  – Euler and Modified Euler methods
  – Runge-Kutta (R-K) Methods
  – Trapezoidal Rule method
• Time-domain simulation for a multi-machine power system
  – Using a simplified DE model (classical generators + impedance loads)
  – Using a general DAE model
Numerical Integration Methods

• The differential equations to be solved in power system stability analysis are nonlinear ODEs (ordinary differential equations) with known initial values $x=x_0$ and $t=t_0$

$$\frac{dx}{dt} = f(x, t)$$

where $x$ is the state vector of $n$ dependent variables and $t$ is the independent variable (time). Our objective is to solve $x$ as a function of $t$

• Explicit Methods
  – In these methods, the value of $x$ at any value of $t$ is computed from the knowledge of the values of $x$ from only the previous time steps, e.g. Euler method and R-K methods

• Implicit Methods
  – These methods use interpolation functions involving future time steps for the expression under the integral, e.g. the Trapezoidal Rule
Euler Method

• The Euler method is equivalent to using the first two terms of the Taylor series about $x$ around the point $(x_0, t_0)$, referred to as a first-order method (error is on the order of $\Delta t^2$)

  - Approximate the curve at $x=x_0$ and $t=t_0$ by its tangent
    \[
    \left. \frac{dx}{dt} \right|_{x_0} = f(x_0, t_0)
    \]

    \[
    \Delta x \approx \left. \frac{dx}{dt} \right|_{x_0} \Delta t \hspace{1cm} x_1 = x_0 + \Delta x = x_0 + \left. \frac{dx}{dt} \right|_{x_0} \Delta t
    \]

  - At step $i+1$
    \[
    x_{i+1} = x_i + \left. \frac{dx}{dt} \right|_{x_i} \Delta t
    \]

• The standard Euler method results in inaccuracies because it uses the derivative only at the beginning of the interval as though it applied throughout the interval
Modified Euler (ME) Method

• Modified Euler method consists of two steps:
  (a) Predictor step:
  \[ x_1^p = x_0 + \frac{dx}{dt}\bigg|_{x_0} \Delta t \]
  \( \Delta t \) is small.
  The derivative at the end of the \( \Delta t \) is estimated using \( x_1^p \)
  \[ \frac{dx}{dt}\bigg|_{x_1^p} = f(x_1^p, t_1) \]
  Estimated slope at the end of \( \Delta t \)

  (b) Corrector step:
  \[ x_1^c = x_0 + \frac{dx}{dt}\bigg|_{x_0} + \frac{dx}{dt}\bigg|_{x_1^p} \Delta t \]
  \[ x_{i+1}^c = x_i + \frac{dx}{dt}\bigg|_{x_i} + \frac{dx}{dt}\bigg|_{x_{i+1}^p} \Delta t \]

• It is a **second-order method** (error is on the order of \( \Delta t^3 \))
• Step size \( \Delta t \) must be small enough to obtain a reasonably accurate solution, but at the same time, large enough to avoid the numerical instability with the computer.
**Runge-Kutta (R-K) Methods**

- General formula of the 2\(^{nd}\) order R-K method:
  (error is on the order of \(\Delta t^3\))
  \[
  k_1 = f(x_0, t_0)\Delta t \\
  k_2 = f(x_0 + \alpha k_1, t_0 + \beta \Delta t)\Delta t \\
  x_1 = x_0 + a_1 k_1 + a_2 k_2
  \]
  At Step \(i+1\):
  \[
  k_1 = f(x_i, t_i)\Delta t = O(\Delta t) \\
  k_2 = f(x_i + \alpha k_1, t_i + \beta \Delta t)\Delta t = O(\Delta t^2) \\
  x_{i+1} = x_i + a_1 k_1 + a_2 k_2
  \]
  - General formula of the 4\(^{th}\) order R-K method:
    (error is on the order of \(\Delta t^5\))
    \[
    x_{i+1} = x_i + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \\
    k_1 = f(x_i, t_i)\Delta t = O(\Delta t) \\
    k_2 = f(x_i + \frac{k_1}{2}, t_i + \frac{\Delta t}{2})\Delta t = O(\Delta t^2) \\
    k_3 = f(x_i + \frac{k_2}{2}, t_i + \frac{\Delta t}{2})\Delta t = O(\Delta t^3) \\
    k_4 = f(x_i + k_3, t_i + \Delta t)\Delta t = O(\Delta t^4)
    \]

The ME method is a special case with \(a_1=a_2=1/2, \alpha=\beta=1\)
Numerical Stability of Explicit Integration Methods

• Numerical stability is related to the **stiffness** of the set of differential equations representing the system.

• The **stiffness** is measured by the ratio of the largest to smallest time constant, or more precisely by $|\lambda_{\text{max}}/\lambda_{\text{min}}|$ of the linearized system.

• **Stiffness** in a transient stability simulation increases with more details (more smaller time constants) being modeled.

• Explicit integration methods have weak stability numerically; with stiff systems, the solution “blows up” unless a small step size is used. Even after the fast modes die out, small time steps continue to be required to maintain numerical stability.
Implicit Methods

- Implicit methods use interpolation functions for the expression under the integral. “Interpolation” implies the function must pass through the yet unknown points at $t_1$
- The simplest implicit integration method is the Trapezoidal Rule method. It uses linear interpolation.
- The stiffness of the system being analyzed affects accuracy but not numerical stability. With larger time steps, high frequency modes and fast transients are filtered out, and the solutions for the slower modes is still accurate. For example, for the Trapezoidal rule, only dynamic modes faster than $f(x_n,t_n)$ and $f(x_{n+1},t_{n+1})$ are neglected.

\[
x_1 = x_0 + \int_{t_0}^{t_1} f(x,t) dt = x_0 + |A| + |B| \approx x_0 + |A|
\]

\[
x_1 = x_0 + \frac{\Delta t}{2} [f(x_0,t_0) + f(x_1,t_1)]
\]

\[
x_{n+1} = x_n + \frac{\Delta t}{2} [f(x_n,t_n) + f(x_{n+1},t_{n+1})]
\]

Compared to ME method:

\[
x_1 = x_0 + \frac{\Delta t}{2} [f(x_0,t_0) + f(x_1^p,t_1)]
\]
Comparison of Explicit and Implicit Methods

\[ \dot{x} = f(x, t) \approx \lambda_{\text{max}} x \]

**Euler Method (explicit)**

\[ x_i = x_{i-1} + f(x_{i-1}, t_{i-1}) \Delta t \]

\[ \approx x_i + \lambda_{\text{max}} x_i \Delta t \]

\[ = x_{i-1} (1 + \lambda_{\text{max}} \Delta t) \]

\[ x_i = x_0 (1 + \lambda_{\text{max}} \Delta t)^i \]

The method is numerically stable if

\[ |1 + \lambda_{\text{max}} \Delta t| < 1 \]

\[ \Leftrightarrow \lambda_{\text{max}} \text{ has a negative real part and} \]

\[ \Delta t < \frac{2}{|\lambda_{\text{max}}|} \]

**Backward Euler Method (implicit)**

\[ x_i = x_{i-1} + f(x_i, t_i) \Delta t \]

\[ \approx x_{i-1} + \lambda_{\text{max}} x_i \Delta t \]

\[ x_i = x_{i-1} \frac{1}{1 - \lambda_{\text{max}} \Delta t} \]

\[ x_i = x_0 \left( \frac{1}{1 - \lambda_{\text{max}} \Delta t} \right)^i \]

\[ \Delta t \text{ can be arbitrarily large as long as} \]

\[ \lambda_{\text{max}} \text{ has a negative real part} \]

(this method has **A-Stability**)

\[ \lambda_{\text{max}} \text{ has a negative real part} \]
• Consider these classic simplifying assumptions:
  – Each synchronous machine is represented by a voltage source $E'$ with constant magnitude $|E'|$ behind $X'_d$ (neglecting armature resistances, the effect of saliency and the changes in flux linkages); The mechanical rotor angle of each machine coincides with the angle of $E'$
  – The governor’s actions are neglected and the input powers $P_{mi}$ are assumed to remain constant during the entire period of simulation
  – Using the pre-fault bus voltages, all loads are converted to equivalent admittances to ground. Those admittances are assumed to remain constant (constant impedance load models)
  – Damping or asynchronous powers are ignored.
  – Machines belonging to the same station swing together and are said to be coherent. A group of coherent machines is represented by one equivalent machine
• Solve the initial power flow and determine the initial bus voltage phasors $V_i$.

• Terminal currents $I_i$ of $m$ generators prior to disturbance are calculated by their terminal voltages $V_i$ and power outputs $S_i$, and then used to calculate $E'_i$

$$I_i = \frac{S_i^*}{V_i^*} = \frac{P_i - jQ_i}{V_i^*}, \quad E'_i = V_i + jX'_{di}I_i \quad i = 1, 2, \ldots, m$$

• All loads are converted to equivalent admittances:

$$y_{i0} = \frac{S_i^*}{|V_i|^2} = \frac{P_i - jQ_i}{|V_i|^2}$$

• To include voltages behind $X'_{di}$, add $m$ internal generator buses to the $n$-bus power system network to form a $n+m$ bus network (ground as the reference for voltages):
Node voltage equation with ground as reference

\[ \mathbf{I}_{bus} = \mathbf{Y}_{bus} \mathbf{V}_{bus} \]

- \( \mathbf{I}_{bus} \) is the vector of the injected bus currents
- \( \mathbf{V}_{bus} \) is the vector of bus voltages measured from the reference node
- \( \mathbf{Y}_{bus} \) is the bus admittance matrix:
  - \( Y_{ii} \) (diagonal element) is the sum of admittances connected to bus \( i \)
  - \( Y_{ij} \) (off-diagonal element) equals the negative of the admittance between buses \( i \) and \( j \)

Compared to the \( \mathbf{Y}_{bus} \) for power flow analysis, additional \( m \) internal generator nodes are added and \( Y_{ii} \) \((i \leq n)\) is modified to include the load admittance at node \( i \)
• To simplify the analysis, all nodes other than the generator internal nodes are eliminated as follows

\[
\begin{bmatrix}
0 \\
I_m
\end{bmatrix} = \begin{bmatrix}
Y_{nn} & Y_{nm} \\
Y_{nm}^t & Y_{mm}
\end{bmatrix} \begin{bmatrix}
V_n \\
E'_m
\end{bmatrix}
\]

\[V_n = -Y_{nn}^{-1}Y_{nm}E'_m\]
\[I_m = [Y_{nm} - Y_{nm}^t Y_{nn}^{-1}Y_{nm}]E'_m = Y_{bus}^{red}E'_m\]

\[S_{ei}^* = E'_{i}^* I_i\quad \text{where} \quad I_i = \sum_{j=1}^{m} E'_j Y_{ij}\]

\[P_{ei} = \Re[E'_{i}^* I_i] = \sum_{j=1}^{m} |E'_i||E'_j||Y_{ij}| \cos(\theta_{ij} - \delta_i + \delta_j)\]

where \(\theta_{ij}\) is the angle of \(Y_{ij}\)

\[2\frac{H_i}{\omega_0} \frac{d^2 \delta_i}{dt^2} = P_{mi} - \sum_{j=1}^{m} |E'_i||E'_j||Y_{ij}| \cos(\theta_{ij} - \delta_i + \delta_j)\]

\[0 = Y_{nn} V_n + Y_{nm} E'_m\]
\[I_m = Y_{nm}^t V_n + Y_{mm} E'_m\]

\[Y_{bus}^{red} = Y_{nm} - Y_{nm}^t Y_{nn}^{-1}Y_{nm}\]

\(Y_{bus}^{red}\) needs to be updated whenever the network is changed.
The overall system equations are expressed in the general form comprising a set of first-order differential equations (dynamic devices) and a set of algebraic equations (devices and network)

\[
\dot{x} = f(x, V) \quad \text{DE}
\]

\[
I(x, V) = Y_N V \quad \text{AE}
\]

where

- \( x \) state vector of the system
- \( V \) bus voltage vector
- \( I \) current injection vector
- \( Y_N \) node admittance matrix. It is constant except for changes introduced by network-switching operations; symmetrical except for dissymmetry introduced by phase-shifting transformers
Solution of the Equations

- Schemes for the solution of equations DE and AE are characterized by the following factors
  - The integration method, i.e. an implicit method or explicit method, used to solve the DE.
  - The method used to solve the AE (power flow analysis), e.g. the Newton-Raphson method.
  - The manner of interface between the DE and AE. Either a partitioned approach or a simultaneous approach may be used

- Most commercialized power system simulation programs provide the Modified Euler, 2nd order R-K, 4th order R-K and Trapezoidal Rule methods
Partitioned Solution

- Partitioned (alternating) solution:
  - to solve $\mathbf{x}$, $\mathbf{V}$ is needed, which may be only approximately available.
  - Similarly, to solve $\mathbf{V}$, $\mathbf{x}$ may not be known accurately.
  - This can lead to interface errors, the elimination of which would require the iteration of the process of solution and extrapolation at each step.

\[ \dot{\mathbf{x}} = f(\mathbf{x}, \mathbf{V}) \]

Only available from the last time step

\[ 0 = \mathbf{I}(\mathbf{x}, \mathbf{y}) - \mathbf{YV} \]

Not exactly equal

Extrapolation:

\[ \mathbf{V}_k \leftarrow \mathbf{V}_{k-1}, \mathbf{V}_{k-2}, \cdots \]

Iteration:

\[ \mathbf{I}(\mathbf{x}, \mathbf{y}) - \mathbf{YV} = \varepsilon \rightarrow 0 \]
A sample approach for partitioned solution with explicit integration

- **DE** and **AE** are solved separately:

  1. At \( t=0^- \), the system is in steady state, initial values of \( \mathbf{x}, \mathbf{V} \) and \( \mathbf{I} \) are known, and \( \mathbf{f}(\mathbf{x}, \mathbf{V})=0 \).

  2. Following a disturbance (e.g. a fault), \( \mathbf{x} \) cannot change instantly. Solve **AE** to give \( \mathbf{V} \) and \( \mathbf{I} \), and the corresponding power flows and other non-state variables of interest at \( t=0^+ \). Then, \( \mathbf{f}(\mathbf{x}, \mathbf{V}) \) can be computed by using any known \( \mathbf{x} \) and \( \mathbf{V} \).

  3. Perform an explicit integration method (say the 2\(^{nd}\) order R-K method) to solve **DE** for each time step of \( \Delta t \): (say step \( t_{n+1} \))

     i. Compute \( k_1 = \mathbf{f}(\mathbf{x}_n, \mathbf{V}_n)\Delta t \) and \( k_2 = \mathbf{f}(\mathbf{x}_n+k_1, \mathbf{V}_n)\Delta t \) at \( t_n \).

     ii. Compute \( \mathbf{x}_{n+1} = \mathbf{x}_n + (k_1+k_2)/2 \)

  4. Solve **AE** to give \( \mathbf{V}_{n+1} \) and \( \mathbf{I}_{n+1} \). Thus all values for \( t_{n+1} \) can be obtained. If the system has a switching operation, the network variables change instantly but not the state variables

- **Advantage**: Programming flexibility, simplicity, reliability and robustness. Since the solution of **DE** requires values only from the previous step, the **DE** associated with each device may be solved independently

- **Disadvantage**: Susceptibility to numerical instability. For a stiff system, a small time step is required throughout the solution period.
Optimal Sequence for Integration \[1\]

- Errors can be reduced if a good sequence is taken for integration of DEs

\[
\frac{dE'_d}{dt} = \frac{1}{T_{do}} \left[-E'_q + (x_d - x'_d)i_d + E_{fd}\right]
\]

\[
\frac{dE'_q}{dt} = \frac{1}{T_{qo}} \left[-E'_d - (x_q - x'_q)i_q\right]
\]

\[
\frac{d\delta}{dt} = \omega_B (S_m - S_{mo})
\]

\[
\frac{dS_m}{dt} = \frac{1}{2H} \left[T_m - DS_m - T_e\right]
\]

\[
T_e = E'_q i_q + E'_d i_d + (x'_d - x'_q)i_d i_q
\]

1. \( \dot{x}_E = [A_E]x_E + [B_E]u_{PSS} + [B_{E2}]V_t \)

2. \( E_{fd} = E_{fd}(x_E) \)

3. \( \dot{x}_T = [A_T]x_T + [B_T]S_m + [B_{T2}]^{pref} \)

4. \( T_m = T_m(x_T) \)

Figure 12.3: Interconnections among subsystems

Simultaneous Solution with Implicit Integration

\[ \dot{x} = f(x, V) \]

\[ 0 = I(x, y) - YV \]

Trapezoidal rule

\[ x_{n+1} = x_n + \frac{\Delta t}{2} [f(x_{n+1}, V_{n+1}) + f(x_n, V_n)] \]

\[ I(x_{n+1}, V_{n+1}) = Y_N V_{n+1} \]

Apply Newton method to solve

\[ F(x_{n+1}, V_{n+1}) = x_{n+1} - x_n - \frac{\Delta t}{2} [f(x_{n+1}, V_{n+1}) + f(x_n, V_n)] = 0 \]

\[ G(x_{n+1}, V_{n+1}) = Y_N V_{n+1} - I(x_{n+1}, V_{n+1}) = 0 \]

\[
\begin{bmatrix}
-F(x_{n+1}^k, V_{n+1}^k) \\
-G(x_{n+1}^k, V_{n+1}^k)
\end{bmatrix} = J
\begin{bmatrix}
\Delta x_{n+1}^k \\
\Delta V_{n+1}^k
\end{bmatrix}
\]

\[
J =
\begin{bmatrix}
\frac{\partial F}{\partial x} & \frac{\partial F}{\partial V} \\
\frac{\partial G}{\partial x} & \frac{\partial G}{\partial V}
\end{bmatrix} =
\begin{bmatrix}
A_D & B_D \\
C_D (Y_N + Y_D)
\end{bmatrix}
\]