Three Phase Linear Tracking State Estimator
Development & Implementation

Kevin D. Jones
kdjones@vt.edu
Outline

• Phase 1
  – Linear State Estimation & Three Phase LSE
  – Topology Processing
  – Matlab Implementation

• Phase 2
  – Migration to C#/openPDC
  – Application Implementation
Linear State Estimation

- Uses PMU measurements exclusively

- Measurement set is bus voltage phasors and line current phasors

- State vector is complex

- Natural evolution of state estimation

- Eliminates the possibility of divergence, scan times

\[
\begin{bmatrix}
V_i \\
V_j \\
I_{ij} \\
I_{ji}
\end{bmatrix} =
\begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
y_{ij} + y_i0 & -y_{ij} & y_{ij} + y_j0
\end{bmatrix}
\begin{bmatrix}
V_i \\
V_j
\end{bmatrix}
\]

\[
[z] = [E] = \begin{bmatrix}
E \\
I
\end{bmatrix} = [\begin{bmatrix}
II \\
yA + y_s
\end{bmatrix}][x] + [e]
\]

\[
[x] = \left[(B^TW^{-1}B)^{-1}B^TW^{-1}\right][z] = [H][z]
\]
Three Phase Linear State Estimation

- Small differences from positive sequence
  - Three phase impedances
  - Matrix formulation

\[
[Y] = \begin{bmatrix}
  y_{1a} & y_{1b} & y_{1c} & 0 & 0 & 0 & \ldots & 0 & 0 & 0 \\
  y_{1b} & y_{1d} & y_{1e} & 0 & 0 & 0 & \ldots & 0 & 0 & 0 \\
  y_{1c} & y_{1e} & y_{1f} & 0 & 0 & 0 & \ldots & 0 & 0 & 0 \\
  y_{2a} & y_{2b} & y_{2c} & & & & \ldots & 0 & 0 & 0 \\
  y_{2b} & y_{2d} & y_{2e} & & & & \ldots & 0 & 0 & 0 \\
  y_{2c} & y_{2e} & y_{2f} & & & & \ldots & 0 & 0 & 0 \\
  \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \ldots & \vdots & \vdots & \vdots \\
  y_{6a} & y_{6b} & y_{6c} & \ldots & y_{6b} & y_{6d} & y_{6e} & \ldots & y_{6c} & y_{6e} & y_{6f} \\
  y_{6b} & y_{6d} & y_{6e} & \ldots & y_{6b} & y_{6d} & y_{6e} & \ldots & y_{6c} & y_{6e} & y_{6f} \\
  y_{6c} & y_{6e} & y_{6f} & \ldots & y_{6c} & y_{6e} & y_{6f} & \ldots & y_{6c} & y_{6e} & y_{6f} \\
\end{bmatrix}
\]

\[
[I] = \begin{bmatrix}
  1 & 0 & 0 \\
  0 & 1 & 0 \\
  0 & 0 & 1
\end{bmatrix}, \quad [0] = \begin{bmatrix}
  0 & 0 & 0 \\
  0 & 0 & 0 \\
  0 & 0 & 0
\end{bmatrix}
\]

\[
Z_{abc} = \begin{bmatrix}
  Z_{aa} & Z_{ab} & Z_{ac} \\
  Z_{ab} & Z_{bb} & Z_{bc} \\
  Z_{ac} & Z_{bc} & Z_{cc}
\end{bmatrix}
\]
Topology Processing

• SCADA information is too slow to use
• Instead, use current flow phasors; bring in breaker statuses in phasor data stream

Outage Criteria
– Apply logical filter to current measurement vector (flow? No flow?)
– No flow indicates potential outage
– Outage is confirmed by breaker statuses & lookup table
– 100% consistency required
– System Matrix is updated
Topology Processing

• Updating the System Matrix after a Contingency
  – Repopulation of system matrix & pseudo-inverse can be cumbersome
  – Method to update pseudo-inverse after line outage

\[ \hat{z} = \left[ H (H^T H)^{-1} H^T \right] z = [Z][z] \]

\[ K(3l - 2: 3l, 3b_l - 2: 3b_l) = [I] \]

\[ K(3m - 2: 3m, 3b_m - 2: 3b_m) = [I] \]

\[ [S] = [K^T ZK] = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & S_1 & 0 & S_3 \\ 0 & 0 & 0 & 0 \\ 0 & S_3 & 0 & S_2 \end{bmatrix} \]

\[ [T] = [I - S]^{-1} = \begin{bmatrix} I & 0 & 0 & 0 \\ 0 & U_1 & 0 & -U_3 \\ 0 & 0 & I & 0 \\ 0 & -U_3 & 0 & U_2 \end{bmatrix}^{-1} \]

\[ T_3 = \left( U_3 - U_2 U_3^{-1} U_1 \right)^{-1} \]

\[ T_1 = T_3 U_2 (U_3)^{-1} \]

\[ T_2 = T_3 U_1 (U_3)^{-1} \]

\[ [M_1] = [M][I - KTK^T Z][I - KKT] \]
Matlab Implementation

- Represents work completed during initial phase of Dominion/DOE project
- For proof-of-concept & initial testing
- How to generate three phase data?
- NDA - IEEE 14 Bus System

\[
\begin{align*}
[I_{unbalanced}] &= [Y_{unbalanced}][V_{balanced}] \\
[I_{balanced}] &= [Y_{balanced}][V_{balanced}] \\
[V_{unbalanced}] &= [Z_{unbalanced}][I_{balanced}] \\
[V_{final}] &= \frac{([V_{balanced}] + [V_{unbalanced}])}{2} \\
[I_{final}] &= \frac{([I_{balanced}] + [I_{unbalanced}])}{2}
\end{align*}
\]
Matlab Results (SE)

**Estimator Output**

- Real (p.u.):
  - 0.371 to 0.377
- Imaginary (p.u.):
  - -0.877 to -0.878

**Measurements & Estimator Output**

- Real (p.u.):
  - 0.345 to 0.395
- Imaginary (p.u.):
  - -0.91 to -0.85

**Error of Real Part of Phasor**

- Real Error (p.u.):
  - 0 to 0.03
- Imaginary (p.u.):
  - 0 to 0.03

**Error of Imaginary Part of Phasor**

- Real (p.u.):
  - 0 to 90
- Imaginary Error (p.u.):
  - 0 to 0.03
- State Variable:
  - 0 to 90
High Level Architecture

- Open source software PDC
- Allows for development of custom phasor concentration code
- Applications called ‘Adapters’, C# libraries
- Extreme Optimizations library for Linear Algebra computations
- 3-φ LSE, Topology Processor, Meter Calibration, Islanding Detection, Unbalanced Condition Monitor
Conclusion & Future Work

• Individual applications successfully migrated to openPDC
• Configuration files simplify changes/maintenance
• Applications to be integrated with each other and tested on realistic data set
• Process repeated on Dominion’s development server
References


