**FLAT START**

**JACOBIAN**

Powerworld Model Jacobian

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

Python Code Jacobian

A screenshot of a computer

Description automatically generated

*As seen between the Powerworld model and my python code, I was able to build a Jacobian with the same values.*

*To build my jacobian matrix, I started by initializing 4 matrices of 0’s for each of the 4 quadrants of the Jacobian. For the J1 matrix, I subtracted a row and column for the slack bus to make a 6x6 matrix. For the J2 matrix, I subtracted a row and column for the slack bus and a column for the PV bus to make a 5x6 matrix. For the J3 matrix, I subtracted a row and column for the slack bus and a row for the PV bus to make a 6x5 matrix. For J4 I subtracted I row and column for the slack and PV bus to make a 5x5 matrix.*

*I then calculated the values for the quadrants of the Jacobian using the below equations:*

*A math equations and formulas

Description automatically generated with medium confidence*

*A math equations and formulas

Description automatically generated with medium confidence*

*After calculating the values for each quadrant of the Jacobian, I used the np.hstack and np.vstack commands to combine them into a single 11x11 matrix.*

**MISMATCH**

Powerworld Model Mismatch

A screenshot of a computer

Description automatically generated

Python Code Mismatch

A screenshot of a computer

Description automatically generated

*As seen above, my calculated mismatch at flat start are equivalent to the mismatch provided by powerworld.*

*To calculate my mismatch I initialized a matrix ‘powerINT’ filled with the input values of the loads for the system. I then utilized the power injection eqns seen below to build a ‘power’ matrix.*

*A math equations with numbers and symbols

Description automatically generated with medium confidence*

*After computing the power injected into the system, I subtracted the power injected matrix from the power initial matrix to get my mismatch values. After completing that, I subtracted voltages for the slack and PV bus and subtracted the angle for the slack bus to develop my mismatch.*

**XVECTOR**

Powerworld Model XVector (Voltages / Angles)

A screenshot of a computer

Description automatically generated

Python Code XVector

A screenshot of a computer

Description automatically generated

*As seen above my values for my solution vector are almost identical to those calculated by powerworld. Of note, my angle values are in radians where powerworld’s angle values are in degrees. These can easily be checked by converting my angles from radians to degrees and they match the angles in powerworld.*

*To calculate my solution vector, I started by making a vector of 0’s and subtracting angles for the slack bus and voltages for the slack and PV bus. I then utilized the ‘np.linalg.solve’ command to cross my mismatch with my jacobian to develop my solution vector.*

**CONVERGENCE**

*It took my code 8 iterations to converge with a tolerance of <0.0001*

**JACOBIAN**

Powerworld Model Jacobian

A screenshot of a computer

Description automatically generatedA screenshot of a computer screen

Description automatically generated

Python Code Jacobian

A screenshot of a computer

Description automatically generated

*This is just to show that as my code converges my jacobian matrix still matches that of powerworld.*

**MISMATCH**

Powerworld Model Mismatch

A screenshot of a computer

Description automatically generated

Python Code Mismatch

A screenshot of a computer

Description automatically generated

*This shows that my code reaches the tolerance across all values for my mismatch.*

**XVECTOR**

Powerworld Model XVector

A screenshot of a computer

Description automatically generated

Python Code XVector

A screenshot of a computer

Description automatically generated

*This shows that my converged values for my voltages and angles match what powerworld’s convergence values are. Once again, my angles are shown in radians in stead of degrees.*

**CURRENT**

Powerworld Model Current

A diagram of a diagram

Description automatically generated

Python Code Current

A screenshot of a computer

Description automatically generated

*As shown from the powerworld model above, my line current values are incredible close in value to what is calculated by powerworld. The differences are minimal.*

*I calculated my line currents by taking the difference in voltage between the two buses connected to the transmission line and dividing it by the impedance of the line. After that, I took the absolute value of the result and multiplied it by 1e3 to get it out of pu.*