# Program Overview

Our program operates by calling a main PowerFlowAnalysis() function for each of the three cases. This function uses the information location to solve for a desired power flow case. Fig. 1 show’s a basic overview of how our power flow analysis program operates. The line data is parsed to create the admittance matrix for the system, and the bus data is turned into a 2D array containing each buses power and voltage information. This 2D matrix is then sent through the power flow process to give each bus’s voltage magnitude and angle, and the real and reactive power.

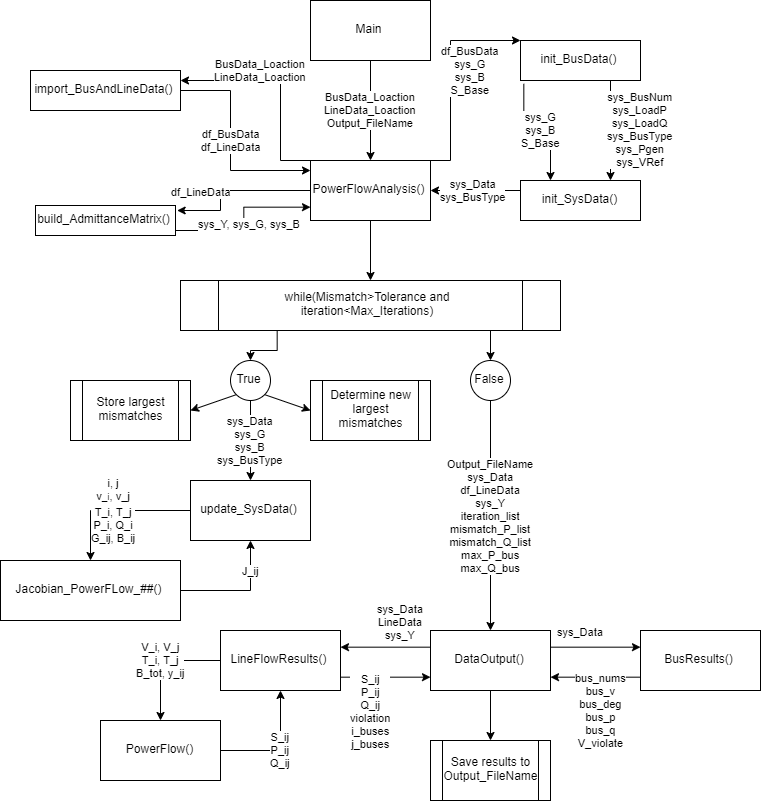


Figure 1: Flow chart of Power Flow Analysis program

The parameters passed to and returned from each function is placed near its corresponding arrow. Important internal processes are shown by process boxes that have stripped left and right borders. A more complete description of each function and variable can be obtained from the code comments in the Appendix.

# Program Validation

A description of the tests that you performed to verify the correctness of your program.

We might also include any missing features/robustness here

* PowerFlow
  + I calculated the current by hand to verify that the current I got from my code equation matched. After I determined that the current matched, I calculated the power by hand and then checked to see if my code output matched the values I got. I then checked to see if those values matched the sample results that are given for the 3-bus example.
* LineFlowResults
  + After assigning each variable a value, I printed the variable to ensure that its value was what I intended for each iteration of the loop. I then checked the resulting powers that I got for inputting those variables into my PowerFlow functions and compared them to the 3-bus sample results. I also checked the line power violation results to see if they also matched with the results from the 3-bus sample problem. I printed out the results of running the function to ensure that the output was what I expected.
* BusResults
  + Like with the LineFlowResults function, I started by printing each variable to make sure they were the values that I intended them to be. I also checked the list of bus voltage violations to see if they matched the 3-bus sample and changed the input values to make sure the if/else loop worked as intended.
* DataOutput
  + Since I was not very familiar with the subfunctions used in this function, I started by checking the format of the created excel spreadsheet and tried changing different inputs to the subfunctions until I obtained a format that fit what I needed. I then created a new excel file after every data addition to ensure that the data was properly being exported to the file.

# Results

## Base Case

V, Theta, P, Q, Violations

P, Q, S in each line, Violations

Convergence record (P, Q Mismatch)

## Contingency 1

V, Theta, P, Q, Violations

P, Q, S in each line, Violations

## Contingency 2

V, Theta, P, Q, Violations

P, Q, S in each line, Violations

## Discussion of Results

Do they make sense? Why?

# Team Contribution

“I helped”

-Eliot Nichols

# Appendix

<http://www.planetb.ca/projects/syntaxHighlighter/popup.php>