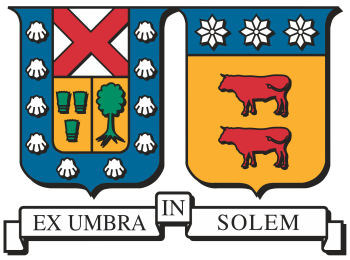

Universidad Técnica Federico Santa María
Departamento de Ingeniería Eléctrica

ANÁLISIS SISTEMAS ELÉCTRICOS DE POTENCIA A

TAREA N°2

Profesor: Guillerno Huerta Soto
Integrantes
Alan Rodríguez Ayala 201623065-3
Sergio Camilo Rojas 201623061-0
Lugar y fecha de entrega: Valparaíso, Viernes 5 de Julio 2024.

Tabla de Contenidos

Figuras

1. Introducción.

Este documento se centra en el análisis de sistemas eléctricos de potencia utilizando pandapower, una herramienta de Python para el análisis de redes eléctricas. El código presentado tiene como objetivo modelar una red eléctrica específica, ejecutar un flujo de carga utilizando el método de Newton-Raphson, a través de la matriz de admitancia nodal (Y-bus). El análisis incluye la creación de una red de buses y líneas, la incorporación de transformadores y generadores, y la simulación de diferentes cargas en la red. La matriz Y-bus es crucial para el análisis de estabilidad y el flujo de potencia en sistemas eléctricos, ya que representa las relaciones entre las distintas barras del sistema. Por lo cual el método de convergencia de Newton-Raphson no solo entrega resultados precisos, sino también permite optimizar el proceso y entender las implicaciones de distintos parámetros lo que nos permite saber como esta operando el SEP.

2. Resumen Ejecutivo.

A continuación se presentan los valores obtenidos en la tarea ocupando el método de convergencia Newton-Raphson para flujos de potencia.

	PandaPower	Algoritmo propio	PandaPower	Algoritmo propio
Barra	Tensiones (V)	Tensiones (V)	Angulos (theta)	Angulos (theta)
1	1,0334	0,44479179	-14,2765	-64,28130994
2	1,0315	0,44479841	-14,4923	-64,28130995
3	1,0324	0,43605521	-14,4698	-64,28092895
4	1,0292	0,44597751	-14,7303	-64,28085122
5	1,0274	0,40546259	-14,8735	-64,27872815
6	1,0284	0,40519895	-14,8649	-64,27383829

Figura 1: Cuadro comparativo entre resultados obtenidos

Según los resultados mostrados en la figura 1 se tienen diferencias significativas entre la librería PandaPower y el método iterativo implementado, las cuales rondan un 56 % entre uno y otro método. Los motivos a esta discrepancias se presentan a profundidad en el apartado de comparativa entre métodos pero se puede mencionar que las principales causales son una mala implementación del método iterativo Newton Raphson, definición de funciones con los parámetros incorrectos, cálculos erróneos de las variables entre otros. Dicho lo anterior y pese a los inconvenientes mencionados el método iterativo de Newton Raphson es una excelente herramienta de resolución para sistemas de flujos de potencia donde se tienen en cuenta varios tipos de variables relacionadas entre si y permite encontrar resultados muy satisfactorios que representan el comportamiento de los sistemas de potencia.

3. Desarrollo de la actividad.

3.1. Arquitectura de sistemas.

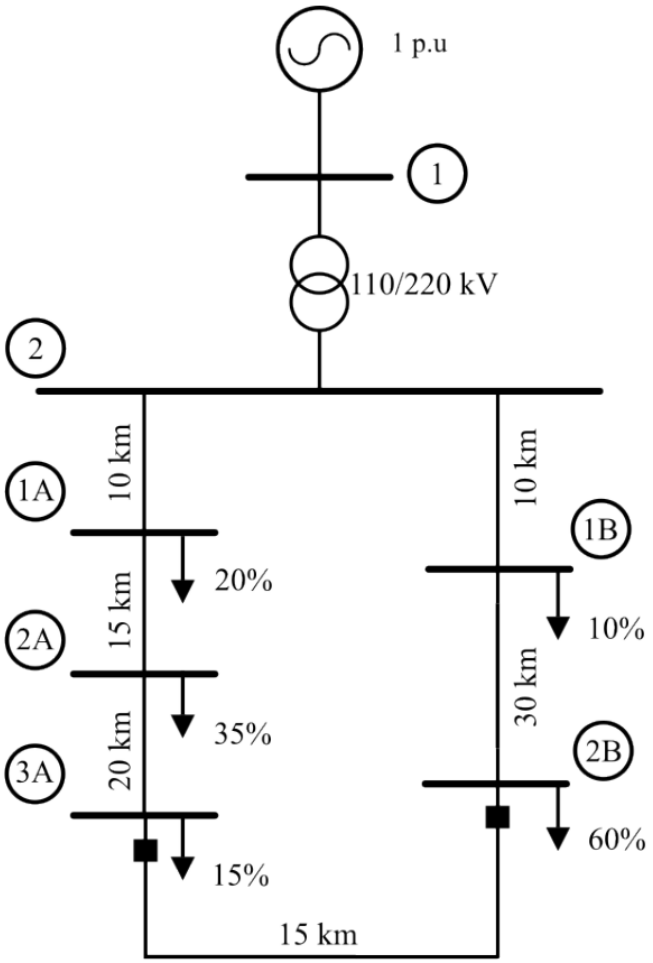

Figura 2: Sistema eléctrico de potencia a analizar

El sistema mostrado en la figura 2 tiene una potencia nominal de 150 [MW] y 100 [MVA] que será la potencia base que se ocupará para el análisis junto con una tensión base de 220 [kV].

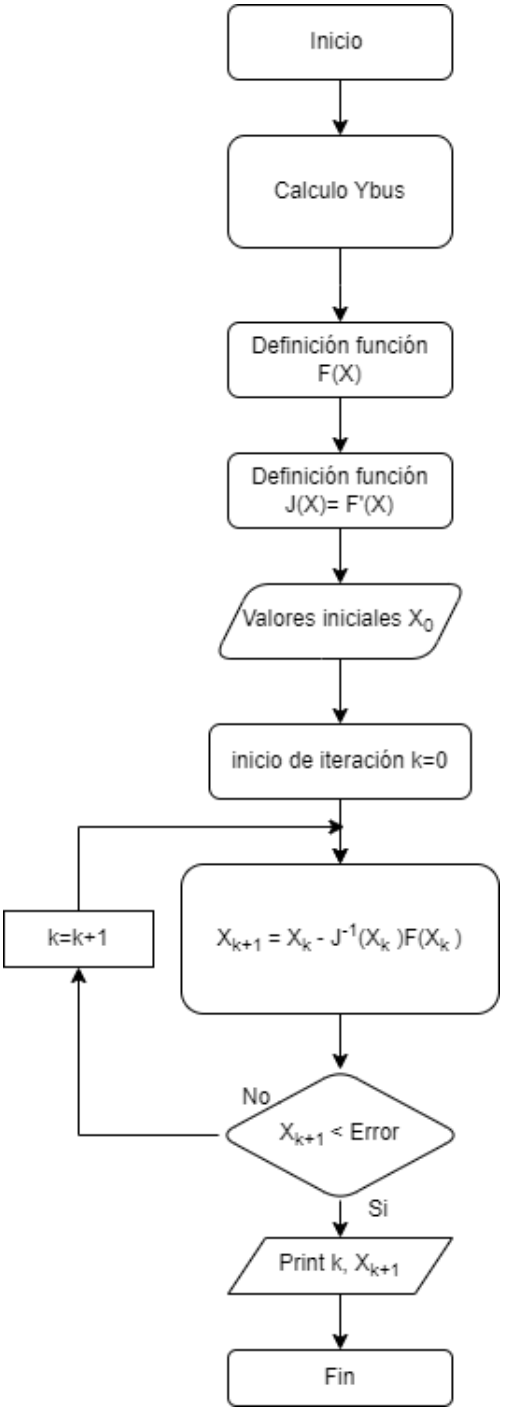

Figura 3: Diagrama de flujo del algoritmo a seguir

Teniendo en consideración la figura 3 se procede a confeccionar el método iterativo Newton Raphson para posteriormente ser comparado con los resultados de la librería pandapower.

3.2. Creación de matriz de admitancias Ybus.

Para la posterior creación del método iterativo Newton Raphson se debe encontrar la matriz de admitancias del sistema Ybus, para ello se ocupa la extensión de la librería pandapower *pandapower.pypower.makeYbus* para que el algoritmo determine los parámetros de la matriz Ybus a través de la siguiente función:

1. Crear la matriz Y-bus
   - Ybus, Yf, Yt = makeYbus(baseZ, bus, branch)
2. Convertir la matriz Y-bus a formato denso
   - Ybus_dense = np.array(Ybus.todense())
3. Convertir la matriz densa a una matriz de NumPy
   - Ybus_np = np.array(Ybus_dense)
4. Aproximar los términos de la matriz a 4 decimales
   - Ybus_rounded = np.round(Ybus_np, decimals=4)

El resultado de la matriz de admitancia Ybus en pu se muestra a continuación:

$$
\begin{aligned}
Y_{11} &= 18,04 - 833,14j \\
Y_{12} &= Y_{21} = -18,04 + 833,14j \\
Y_{22} &= 28090,8 - 4550.01j \\
Y_{23} &= Y_{32} = -14036,38 + 22117,93j \\
Y_{24} &= Y_{42} = -14036,38 + 22117,93j \\
Y_{33} &= 23393,71 - 36839,46j \\
Y_{35} &= Y_{53} = -9357,59 + 14745,29j \\
Y_{44} &= 18715,18 - 29452,57J \\
Y_{46} &= Y_{64} = -4678,79 + 7372,64J \\
Y_{55} &= 16375,77 - 25770,99j \\
Y_{57} &= -7018,19 + 11058,96j \\
Y_{66} &= 14036,38 - 22075,17j \\
Y_{67} &= -9357,59 + 14745,29j \\
Y_{77} &= 16375,78 - 25770,99j
\end{aligned}
$$

La combinación de admitancias no mostradas tienen un valor de 0.

3.3. Definición y confección de la matriz F(x).

$$F(X) = \begin{bmatrix} \Delta P_i \\ \Delta Q_i \end{bmatrix} \tag{3.1}$$

con:

- $\Delta P_i = P_i^* - P_i, P_i^*$:potencia inyectada

$$P_i = V_i \sum_{j=1} V_j (G_{ij} cos(\theta_{ij}) + B_{ij} sin(\theta_{ij})$$

- $\Delta Q_i = Q_i^* - Q_i, Q_i^*$:potencia inyectada

$$Q_i = V_i \sum_{j=1} V_j (G_{ij} sen(\theta_{ij}) - B_{ij} cos(\theta_{ij})$$

3.4. Definición y confección de la matriz Jacobiana J(x).

$$J(X) = \begin{bmatrix} \frac{\delta \Delta P}{\delta \theta} & \frac{\delta \Delta P}{\delta V} \\ \frac{\delta \Delta Q}{\delta \theta} & \frac{\delta \Delta Q}{\delta V} \end{bmatrix} \tag{3.2}$$

$$J(X) = \begin{bmatrix} H & N \\ M & L \end{bmatrix} \tag{3.3}$$

con:

$$H_{ij} = \frac{\delta \Delta P_i}{\delta \theta_j} = -V_i V_j (G_{ij} sen(\theta_{ij}) - B_{ij} cos(\theta_{ij})$$
$$H_{ii} = \frac{\delta \Delta P_i}{\delta \theta_i} = Q_i + B_{ii} V_i^2$$
$$N_{ij} = \frac{\delta \Delta P_i}{\delta V_j} = -V_i (G_{ij} cos(\theta_{ij}) + B_{ij} sen(\theta_{ij})$$
$$N_{ii} = \frac{\delta \Delta P_i}{\delta \theta_i} = -\frac{P_i}{V_i} - G_{ii} V_i$$
$$M_{ij} = \frac{\delta \Delta Q_i}{\delta \theta_j} = V_i V_j (G_{ij} cos(\theta_{ij}) + B_{ij} sen(\theta_{ij})$$
$$M_{ii} = \frac{\delta \Delta Q_i}{\delta \theta_i} = -P_i + G_{ii} V_i^2$$
$$L_{ij} = \frac{\delta \Delta P_i}{\delta \theta_j} = -V_i (G_{ij} sen(\theta_{ij}) - B_{ij} cos(\theta_{ij})$$
$$L_{ii} = \frac{\delta \Delta P_i}{\delta \theta_i} = -\frac{Q_i}{V_i} + B_{ii} V_i$$

3.5. Definición y confección de la matriz de incógnitas X.

Las incógnitas del sistema están en la matriz X :

$$X = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \\ \theta_5 \\ \theta_6 \\ V_1 \\ V_2 \\ V_3 \\ V_4 \\ V_5 \\ V_6 \end{bmatrix} \tag{3.4}$$

con los valores iniciales X_0

$$X_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \tag{3.5}$$

La ecuación a resolver queda expresada como:

$$X_{k+1} = X_k - J^{-1}(X_k) \cdot F(X_k)$$

Luego evaluamos X_0 en F(X) y J(X), se define el Error en $1e^{-3}$, comienza la primera iteración (k=0) quedado así :

$$X_1 = X_0 - J^{-1}(X_0) \cdot F(X_0)$$

3.6. Comparación entre algoritmo y función de librería

Se hará una comparación de resultados entre la función de Newton-Raphson de la librería de PandaPower y el algoritmo creado anteriormente.
Los resultados entregados por la función de librería son los siguientes:

```
Resultados de pandapower:
   vm_pu  va_degree      p_mw  q_mvar
0 1.0000     0.0000 -212.3102  5.8073
1 1.0334   -14.2765    0.0000  0.0000
2 1.0315   -14.4923   30.0000 20.0000
3 1.0324   -14.4698   15.0000 10.0000
4 1.0292   -14.7303   52.5000 35.0000
5 1.0274   -14.8735   90.0000 60.0000
6 1.0284   -14.8649   22.5000 15.0000
```

Figura 4: Resultados obtenidos por librería de PandaPower

Resultados entregados por algoritmo realizado:

Barra	Tensiones (V)	Angulos (theta)
1	0,44479179	-64,28130994
2	0,44479841	-64,28130995
3	0,43605521	-64,28092895
4	0,44597751	-64,28085122
5	0,40546259	-64,27872815
6	0,40519895	-64,27383829

Figura 5: Resultados obtenidos por algoritmo

Como se puede notar a simple vista los valores entregados por el algoritmo tienen una alta diferencia con el que entrega la función de PandaPower.
Los resultados de PandaPower se pueden tomar como referencias para el algoritmo propio, ya que estos son mas precisos.

Esta diferencia se podrían dar por distintos motivos, uno de ellos puede ser que al implementar el algoritmo propio tenga errores al calcular las funciones utilizadas en el método, con esto se ve afectada la convergencia y los resultados finales.

Cabe destacar que el método iterativo no converge a un valor, por lo que no se pueden ver las iteraciones realizada por él. el método iterativo solo se mantiene en un ciclo en torno a los valores entregados,

Para solucionar dicho inconveniente se propone rehacer el código iterativo o comparar valores de cada iteración con resultados matemáticos calculados manualmente, no se presenta dicha comparación debido a la poca disponibilidad de tiempo.

En cuanto a lo esperable según el nivel de tolerancia que se imponga en el método iterativo de Newton Raphson, al tener un valor numérico de tolerancia más pequeño, se espera que el método converja en un mayor numero de iteraciones lo cual supone una mayor carga para el programa, pero se tendrá un resultado más preciso sobre las variables del sistema. En cambio, al aumentar el valor numérico de la

tolerancia, se tendrá que en un menor número de iteraciones el método converja a un resultado, siendo menos demandante para el programa pero teniendo una menor precisión sobre los resultados de las variables del sistema.

En el caso del sistema analizado se considera que un óptimo margen de tolerancia se encuentre entre los rangos de 10^{-6} a 10^{-3} debido a que se trabaja en valores de por unidad, tolerancias más pequeñas agregarían muchas más iteraciones que ganancias en la precisión de los resultados, en cuanto a valores de tolerancia mayores al mencionado agregarían mayor imprecisión de los resultados que ganancia en la rapidez de convergencia del método iterativo.

4. Conclusiones.

La aplicación del método de Newton-Raphson, tanto a través del algoritmo desarrollado en el presente trabajo como mediante la librería PandaPower están basados en el mismo principio teórico, al ejecutar ambos métodos y compararlos se mostró diferencias significativas en su implementación lo que resultan en distintas variaciones en los resultados obtenidos. Se debe realizar una revisión y realizar posibles modificaciones del algoritmo iterativo para mejorar su capacidad de convergencia, al igual que realizar comparaciones iterativas con resultados calculados manualmente para validar la precisión del algoritmo en sus primeras iteraciones y también el poder describir de buena manera los parámetros utilizados en el algoritmo propio. Pese a estos resultados dispares debido muy posiblemente a lo antes mencionado, se tiene este este método es una gran herramienta en la resolución de sistemas complejos.

Mencionado lo anterior se tiene que al utilizar el método de Newton Raphson permite realizar un detallado estudio de un SEP. Al obtener una convergencia, es posible determinar flujos de potencia, La mayoria de las veces es necesario realizar más de una iteración en el método para lograr los valores de convergencia y conocer las incógnitas del sistema a través del diseño de un código programado para poder realizar el método, debido a la complejidad de los cálculos, mencionado lo anterior se tiene que este método ayuda a encontrar soluciones con relativa eficiencia y rapidez en comparativa con otros métodos existentes lo que genera una ventaja comparativa cuando los sistemas a analizar son complejos en cantidad de variables existentes. Además, el método Newton Rapshon ayuda a la identificación de anormalidades en el funcionamiento del sistema, pudiendo así orientar a la búsqueda de soluciones óptimas para obtener un correcto funcionamiento.