CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA CELSO SUCKOW
DA FONSECA - CAMPUS NOVA FRIBURGO
ENGENHARIA ELÉTRICA

Análise de Sistemas de Potência

Projeto 3 – Solução do Fluxo de Potência no Sistema IEEE 14 Barras via Algoritmo de Newton-Raphson em Python

Semestre: 2°/2025

Sergio Pedro Rodrigues Oliveira

Prof. Thiago Resende de Almeida

Nova Friburgo
2025

Sumário

Lista de Figuras

1 Introdução

O setor de **Análise de Sistemas de Potência (ASP)** exige ferramentas computacionais robustas e precisas para garantir o planejamento, a operação e a confiabilidade das redes elétricas. Dentre as várias análises realizadas, o estudo de **Fluxo de Potência (FP)** é o fundamental, pois determina o estado de regime permanente do sistema em condições específicas de geração e carga.

O método mais amplamente empregado e numericamente eficiente para resolver estas equações é o **Método de Newton-Raphson (MNR)**, conhecido pela sua rápida convergência quadrática.

O **objetivo central** deste trabalho é a implementação e validação computacional do **Método de Newton-Raphson em Python** para a análise de um sistema de potência de referência. Este projeto constitui um requisito prático da disciplina de **Análise de Sistemas de Potência** da Faculdade de Engenharia Elétrica do **CEFET/RJ - Campus Nova Friburgo**, visando demonstrar a compreensão das formulações matemáticas e a aplicação de técnicas de programação modular e de testes unitários. O programa será desenvolvido de forma modular, com foco na precisão da álgebra matricial e na robustez da convergência, e será aplicado ao sistema padrão **IEEE 14 Barras**. Os resultados obtidos serão validados por meio de comparação direta com uma análise de referência gerada pelo software **ANAREDE**, assegurando a exatidão do código desenvolvido.

Figura 1: Diagrama Unifilar do Sistema de Teste IEEE 14 Barras.

2 Etapas de Desenvolvimento do Projeto

O desenvolvimento do programa de Fluxo de Potência seguiu uma abordagem estruturada e modular, garantindo que cada componente fosse verificado antes da integração no ciclo iterativo principal.

2.1 Cronograma de Desenvolvimento

O projeto foi dividido em tarefas objetivas e com prazos definidos, conforme a metodologia ágil de execução. A Tabela 1 detalha a sequência das tarefas, incluindo a precedência entre elas.

Tabela 1: Cronograma Detalhado do Projeto de Fluxo de Potência (IEEE 14 Barras)

Tarefa	**Dias**	**Início**	**Prazo Final**	**Precedência**
Fase 1: Preparação (Até 28/11/2025)				
1. Planejamento (Concluído)	6	19/11	24/11	-
2. Tratamento dos Dados (p.u.)	2	24/11	25/11	**1**
3. Diagrama Unifilar	5	24/11	28/11	**1**
4. Matriz $\mathbf{Y_{bus}}$	3	26/11	28/11	**2**
Fase 2: Implementação do Fluxo de Potência (29/11/2025 a 06/12/2025)				
5.1. Chute Inicial	1	29/11	29/11	**4**
5.2. Vars. de Estado e Mismatch	1	30/11	30/11	**5.1**
5.3. Eq. Potência (P e Q)	1	01/12	01/12	**5.2**
5.4. Matriz Jacobiana (**J**)	2	01/12	02/12	**4; 5.2**
5.5. Solução Sistema Linear	2	03/12	04/12	**5.4**
5.6. Resolver Eq. Sub-Sistema 2	2	05/12	06/12	**5.5**
Entrega Final				
Submissão do Programa	—	—	07/12	—

O desenvolvimento seguiu a seguinte sequência de passos:

1. **Planejamento Teórico e Estrutura Inicial:** Revisão da Fundamentação Teórica (Equações de P e Q, Método de Newton-Raphson). Definição da Base de Potência (**100** MVA) e do Sistema de Teste (**IEEE 14 Barras**).

2. **Tratamento dos Dados (p.u.) e Diagrama Unifilar:** Implementação da rotina de leitura do arquivo **TXT tabelado** e utilização da biblioteca **Pandas**. Conversão de todos os dados de Potência e Impedância para unidades

por unidade (**p.u.**), e criação paralela do **Diagrama Unifilar** do sistema.

3. **Desenvolvimento da Matriz de Admitância ($\mathbf{Y_{bus}}$):** Criação da função para montagem da Matriz de Admitância, incluindo o tratamento adequado de Tap nos transformadores (cujo Tap $\neq 1.0$ p.u.).

4. **Implementação do Núcleo Iterativo (Sub-sistema 1):** O desenvolvimento modular do algoritmo de Newton-Raphson seguiu as sub-etapas 5.1 a 5.5, culminando na construção do **Loop Principal** e na solução iterativa do sistema linear para a atualização das **variáveis de estado (θ e V)**.

5. **Cálculo Pós-Convergência (Sub-sistema 2):** A etapa final de codificação (sub-etapa 5.6) envolveu o cálculo das potências desconhecidas (P_{Slack} e Q_{PV}), completando as equações necessárias para o resultado final do **Sub-sistema 2**.

2.2 Consultas com o Professor

As consultas com o professor para alinhamento e revisão de etapas-chave foram agendadas nos seguintes marcos:

- **Consulta 1 (Foco: Dados e Estrutura):** 24 de Novembro de 2025
- **Consulta 2 (Foco: Matriz $\mathbf{Y_{bus}}$ e Resultados Parciais):** 01 de Dezembro de 2025

3 Backlog: Algoritmo Newton-Raphson (FLUXO)

3.1 Fase 1: Inicialização e Setup

Tarefa	Descrição
1.1	**Definir Classificação de Barras:** Identificar e listar os **índices** (base zero) das barras **Slack**, **PV** e **PQ**.
1.2	**Definir Ordem dos Estados:** Estabelecer a sequência de correção ($\mathbf{\Delta}\delta$ e $\mathbf{\Delta V}$) no vetor de estados, seguindo a ordem: PV ($\Delta\delta$), PQ ($\Delta\delta$), PQ (ΔV). Esta ordem definirá a estrutura da Jacobiana.
1.3	**Chute Inicial:** Criar os vetores iniciais de módulo de tensão ($\mathbf{V}$) e ângulo (δ), usando os valores de entrada (‘df_barras‘) e chutes iniciais (geralmente $\delta = 0, V = 1.0$ para barras PQ).
1.4	**Vetor de Potências Especificadas:** Calcular e montar o vetor $\mathbf{P}_{\text{esp}}$ e $\mathbf{Q}_{\text{esp}}$ (injeção $P_G - P_L$ e $Q_G - Q_L$) na mesma ordem estabelecida na Tarefa 1.2.

3.2 Fase 2: Iteração Principal (O Loop)

Tarefa	Descrição
2.1	**Função $\mathbf{P_{calc}}$ e $\mathbf{Q_{calc}}$:** Implementar as equações de potência ativa e reativa *calculadas* ($P_{\text{calc}}, Q_{\text{calc}}$) usando os valores atuais de $\mathbf{V}$, δ e a matriz $\mathbf{Y}_{\text{bus}}$.
2.2	**Vetor de Desalinho:** Calcular o vetor $\mathbf{\Delta f}$ (o erro), que é a diferença entre as potências especificadas e as calculadas ($\mathbf{\Delta P} = \mathbf{P}_{\text{esp}} - \mathbf{P}_{\text{calc}}$ e $\mathbf{\Delta Q} = \mathbf{Q}_{\text{esp}} - \mathbf{Q}_{\text{calc}}$).
2.3	**Teste de Convergência:** Verificar se o valor absoluto máximo do vetor de desalinho ($\mathbf{\Delta P}$ e $\mathbf{\Delta Q}$) é menor que a tolerância (ϵ).
2.4	**Montagem da Jacobiana:** Calcular e montar a **Matriz Jacobiana** ($\mathbf{J}$), composta pelas quatro submatrizes de derivadas parciais.
2.5	**Solução do Sistema:** Resolver o sistema linear $\mathbf{J} \cdot \mathbf{\Delta x} = \mathbf{\Delta f}$ para obter o vetor de correção ($\mathbf{\Delta}\delta$ e $\mathbf{\Delta V}$).
2.6	**Atualização de Estados:** Corrigir os vetores $\mathbf{V}$ e δ usando a fórmula $\mathbf{x}^{(i+1)} = \mathbf{x}^{(i)} + \mathbf{\Delta x}$.
2.7	**Verificação de Limites Q:** Para barras PV, verificar se o Q_{calc} está dentro dos limites. Se violado, alterar o tipo da barra de PV para PQ.

3.3 Fase 3: Resultados

Tarefa	Descrição
3.1	**Cálculo da Slack:** Calcular as potências $P_{G,\text{Slack}}$ e $Q_{G,\text{Slack}}$ (injeção líquida) após a convergência.
3.2	**Resultados das Linhas:** Calcular os fluxos de potência, perdas e correntes em todas as linhas.
3.3	**Relatório:** Formatar e apresentar os resultados finais (tensão, ângulo, P, Q, perdas).

4 Metodologia

4.1 Estrutura de Funcionalidades do Programa

O programa foi desenvolvido em três módulos principais que refletem o fluxo de execução do algoritmo, garantindo modularidade, testes unitários e clareza no desenvolvimento.

1. **Módulo de Pré-processamento e Inicialização:**
    - Leitura do arquivo de dados e conversão para unidades por unidade (**p.u.**).
    - Inicialização das variáveis de estado (ângulos $\delta = 0$, tensões $V = 1.0$ p.u. ou especificadas).
2. **Módulo do Núcleo Iterativo (Sub-sistema 1):** Esta fase contém o loop principal do Método de Newton-Raphson e é responsável pela convergência das 22 variáveis de estado (δ e V).
    - Funções de montagem da Matriz de Admitância ($\mathbf{Y_{bus}}$).
    - Funções de cálculo das Injeções de Potência (P_{calc} e Q_{calc}).
    - Funções de cálculo do Vetor Mismatch ($\Delta\mathbf{F}$).
    - Funções de construção da Matriz Jacobiana ($\mathbf{J}$) de 22×22.
    - Rotina de Solução do Sistema Linear ($\Delta\mathbf{x} = \mathbf{J}^{-1}\Delta\mathbf{F}$) e atualização das variáveis.
3. **Módulo de Pós-processamento e Resultados (Sub-sistema 2):** Esta fase é executada após a convergência do Sub-sistema 1.
    - Cálculo das potências desconhecidas (P_{Slack} e Q_{PV}) com os valores finais de δ e V.

- Cálculo dos fluxos de potência ativa e reativa em todas as linhas.
- Geração do relatório final de resultados, incluindo a comparação com o software de referência (**ANAREDE**).

4.2 Sistema de Teste e Dados de Entrada (14 Barras)

O programa foi desenvolvido e validado utilizando o sistema de referência **IEEE 14 Barras**. A Matriz Jacobiana (**J**) terá uma dimensão de $\mathbf{22 \times 22}$, exigindo o uso eficiente das capacidades de álgebra linear do **NumPy**. Os dados de entrada serão lidos de um **documento TXT tabelado** ('IEEE14_novo.txt'). A **Barra Slack** foi definida como a Barra **1**, com tensão especificada de $V_{Slack} = 1.0\angle 0°$ p.u.

4.3 Tipos de Barras e Variáveis de Estado

A formulação do problema de Fluxo de Potência é determinada pelo tipo de cada barra:

- **Barra Slack (Referência ou Swing Bus):** Variáveis Especificadas: Tensão (**V**) e Ângulo (δ).
- **Barra PV (Geração ou Tensão Controlada):** Variáveis Especificadas: Potência Ativa (**P**) e Tensão (**V**).
- **Barra PQ (Carga ou Load Bus):** Variáveis Especificadas: Potência Ativa (**P**) e Potência Reativa (**Q**).

4.4 Critérios de Parada e Convergência

O algoritmo deve ser executado até que um dos dois critérios de parada mutuamente exclusivos seja atendido:

- **Tolerância de Mismatch (ϵ):** Critério de convergência: $\max(|\Delta P|, |\Delta Q|) \leq \mathbf{10^{-6}}$ p.u.
- **Limite Máximo de Iterações:** Limite máximo: **15 iterações**.

4.5 Estrutura, Bibliotecas Python e Estratégia de Testes

- **NumPy:** Essencial para a Álgebra Linear (montagem de $\mathbf{Y_{bus}}$ e solução de $\mathbf{J}$).
- **Pandas:** Recomendada para o **parsing do arquivo TXT tabelado**.
- **cmath:** Necessária para lidar com **números complexos**.
- **unittest:** Biblioteca **nativa de testes** para garantir a integridade modular.

5 Fundamentação Teórica

5.1 O Método de Newton-Raphson (Sumário)

O MNR lineariza o sistema de equações não lineares ($F(x) = 0$). O sistema linear resolvido a cada passo é dado por:

$$\mathbf{J}^{(k)}\Delta\mathbf{x}^{(k)} = \Delta\mathbf{F}^{(k)} \tag{1}$$

As variáveis de estado são atualizadas por:

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \Delta\mathbf{x}^{(k)} \tag{2}$$

5.2 Equações do Fluxo de Potência e o Mismatch

A injeção de **potência complexa líquida** ($\mathbf{S_k}$) na barra **k** é a soma das potências ativa ($\mathbf{P_k}$) e reativa ($\mathbf{Q_k}$):

$$S_k = P_k + jQ_k \tag{3}$$

As componentes ativa e reativa são calculadas a partir das tensões e ângulos atuais ($\mathbf{V_k}$, $\delta_\mathbf{k}$) e dos parâmetros de rede, onde a soma é realizada sobre todas as barras **m** do sistema (N):

$$P_k = V_k \sum_{m=1}^{N} V_m(G_{km}\cos(\delta_k - \delta_m) + B_{km}\sin(\delta_k - \delta_m)) \tag{4}$$

$$Q_k = V_k \sum_{m=1}^{N} V_m(G_{km}\sin(\delta_k - \delta_m) - B_{km}\cos(\delta_k - \delta_m)) \tag{5}$$

O **Vetor Mismatch** ($\Delta\mathbf{F}$) para a barra k é calculado pela diferença entre os valores especificados e os calculados: $\Delta P_k = P_{k,esp} - P_{k,calc}$ e $\Delta Q_k = Q_{k,esp} - Q_{k,calc}$.

5.3 Variáveis de Estado e Particionamento do Sistema

O **Vetor de Variáveis de Estado** a ser corrigido em cada iteração, $\Delta \mathbf{x}^{(k)}$, é composto pelas variáveis que são *desconhecidas* em cada tipo de barra (excluindo a Slack).

O sistema **IEEE 14 Barras** possui 14 barras, das quais:

- 1 Barra Slack (Barra 1): δ e V são conhecidos (fixos).
- 4 Barras PV (Geração): V é conhecido (fixo), δ é desconhecido.
- 9 Barras PQ (Carga): V e δ são desconhecidos.

O **Vetor de Variáveis de Estado** a ser corrigido em cada iteração é composto pelas 22 variáveis desconhecidas (ângulos δ e magnitudes de tensão V) do sistema IEEE 14 Barras. A solução do problema no projeto foi estruturada em duas fases principais, referidas como Sub-sistemas:

- **Sub-sistema 1 (Fase de Correção Iterativa):** Refere-se à solução simultânea do sistema de 22 equações do Método de Newton-Raphson. Esta fase calcula, em cada iteração, as correções para as 13 variáveis de ângulo ($\Delta\delta$) e as 9 variáveis de tensão ($\Delta V/V$), atualizando o estado do sistema até a convergência.
- **Sub-sistema 2 (Fase de Cálculo Pós-Convergência):** Refere-se à etapa subsequente à convergência do Sub-sistema 1, onde são calculadas as grandezas não especificadas, como as injeções de Potência Reativa (Q) nas barras PV e as injeções de Potência Ativa e Reativa na Barra Slack (P e Q).

O número total de variáveis de estado resolvidas no Sub-sistema 1 é, portanto, **22**: 13 variáveis de ângulo (todas as barras, exceto a Slack) e 9 variáveis de tensão (apenas barras PQ). O termo $\Delta V/V$ é adotado no lugar de ΔV para padronizar

a formulação e garantir maior simetria com as equações $\Delta\delta$, o que é uma prática comum na formulação do MNR.

5.4 Construção da Matriz Jacobiana Desejada (Redução por Omissão)

A matriz Jacobiana final **J** ($\mathbf{22 \times 22}$) é obtida através da **omissão** (particionamento) das linhas e colunas relativas às variáveis conhecidas (Slack δ, V e PV V).

5.5 Construção da Matriz Jacobiana Desejada (Redução por Omissão)

A matriz Jacobiana final **J** ($\mathbf{22 \times 22}$) é obtida através da **omissão** (particionamento) das linhas e colunas relativas às variáveis conhecidas (Slack δ, V e PV V).

5.6 A Técnica do "Big M"(Em Implementações Alternativas)

É tecnicamente possível usar a técnica do **"Big M"** para construir a Matriz Jacobiana e impor as restrições de variáveis conhecidas (fixando o erro em zero). Contudo, a técnica de **Omissão** é a abordagem mais robusta e estável.

6 Resultados e Análise

6.1 Validação e Comparação com o ANAREDE

A validação será realizada através da **comparação direta** dos resultados do código Python com os resultados de referência gerados pelo software **ANAREDE**.

- Foco em **Resultados de Barra** (**V** e δ) e **Resultados de Linha** (Fluxos e Perdas Totais).
- O programa será considerado validado se a diferença máxima entre os resultados for **menor que** $\epsilon = 10^{-6}$ **p.u.**.

Tabela de Resultados de Barra (Placeholder)

A tabela a seguir apresenta os resultados finais de tensão e ângulo de fase para cada barra do sistema IEEE 14 Barras, comparados com a referência ANAREDE.

Tabela 2: Resultados Finais do Fluxo de Potência (V e δ em p.u. / rad)

Barra	**Resultados Python**		**Resultados ANAREDE**		**Diferença Máx.**
	$V_{p.u.}$	δ_{rad}	$V_{p.u.}$	δ_{rad}	
1	1.0000	0.0000	1.0000	0.0000	0
2	[Inserir]	[Inserir]	[Inserir]	[Inserir]	[Inserir]
3	[Inserir]	[Inserir]	[Inserir]	[Inserir]	[Inserir]

6.2 Análise da Convergência

Será apresentada uma discussão sobre o número exato de iterações necessárias e a análise do decaimento do vetor Mismatch máximo a cada iteração, comprovando a **convergência quadrática**.

7 Conclusão

O projeto atingiu seu objetivo principal ao implementar o **Método de Newton-Raphson (MNR)** para a solução do Fluxo de Potência (FP) do sistema **IEEE 14 Barras** utilizando o ambiente **Python**.

O algoritmo demonstrou robustez e eficiência, convergindo a solução em **[Inserir Nº de Iterações]** iterações com um Mismatch máximo inferior à tolerância de $\mathbf{10^{-6}}$ p.u.

A **validação** com o software de referência **ANAREDE** atestou a precisão da implementação, especialmente na construção da Matriz $\mathbf{Y_{bus}}$ e na formulação da Matriz Jacobiana particionada. O uso de **'unittest'** garantiu a integridade modular das funções de cálculo.

Referências Bibliográficas

Referências

[1] Ruggiero, M. A. G., and Lopes, V. L. d. R. *Cálculo Numérico: Aspectos Teóricos e Computacionais*, 2ª ed. Makron Books, 1997.

[2] Stevenson, W. D. *Elementos de análise de sistemas de potência*. McGraw-Hill do Brasil, 1974.

A Anexo I: Interpretação do Arquivo de Dados IEEE14_novo.txt

Premissas e Classificação

- Base de Potência: $\mathbf{S_{Base}} = \mathbf{100}$ MVA.
- Classificação das Barras: 1 Slack (**Barra 1**), 4 PV (**Barras 2, 3, 6 e 8**), e 9 PQ (**Demais**).
- Dados de Linha: Incluem transformadores com Tap $\neq 1.0$ (linhas **4-7, 4-9 e 5-6**).

A.1 O Sistema Por Unidade (p.u.) e S_{Base}

O sistema por unidade (p.u.) é utilizado para normalizar as grandezas do sistema. A base de potência foi estabelecida em $\mathbf{S_{Base}} = \mathbf{100}$ MVA. O valor real de qualquer potência deve ser dividido por este valor de base para ser convertido para p.u.:

$$X_{p.u.} = \frac{X_{Real}}{X_{Base}} \tag{6}$$

Para o **cálculo do vetor Mismatch** ($\Delta\mathbf{F}$), é necessário garantir que todas as potências estejam em p.u.:

- **Potência Ativa (P):** O valor em MW (potência real) deve ser dividido por $S_{Base} = 100$ MVA.
- **Potência Reativa (Q):** O valor em Mvar (potência reativa) deve ser dividido por $S_{Base} = 100$ MVA.

Os valores de impedância (R e X) no arquivo de entrada são **assumidos como já estando em p.u.**.