

Desenvolvimento de Interface Gráfica para Estimador de Estado Harmônico aplicado a Sistemas de Distribuição via Linguagem de Programação Python

Andrey Gabriel Rodrigues Pimentel
Instituto de Engenharia, Ciência e
Tecnologia (IECT)
Universidade Federal dos Vales do
Jequitinhonha e Mucuri (UFVJM)
Janaúba-MG, Brasil
andrey.gabriel@ufvjm.edu.br

João Vítor Gomes de Araújo
Escola de Engenharia de São Carlos
(EESC)
Universidade de São Paulo (USP)
São Carlos-SP, Brasil
jv.araujo@usp.br

Cleiton Rocha Sudré
Escola de Engenharia de São Carlos
(EESC)
Universidade de São Paulo (USP)
São Carlos-SP, Brasil
cleiton.sudre@usp.br

Thaís de Fátima Araújo Silva
Instituto de Engenharia, Ciência e
Tecnologia (IECT)
Universidade Federal dos Vales do
Jequitinhonha e Mucuri (UFVJM)
Janaúba-MG, Brasil
thais.araujo@ufvjm.edu.br

Júlia Oliveira Fernandes
Escola de Engenharia de São Carlos
(EESC)
Universidade de São Paulo (USP)
São Carlos-SP, Brasil
juolivf@usp.br

Jáder Fernando Dias Breda
Instituto de Engenharia, Ciência e
Tecnologia (IECT)
Universidade Federal dos Vales do
Jequitinhonha e Mucuri (UFVJM)
Janaúba-MG, Brasil
jader.breda@ufvjm.edu.br

Resumo— O uso de softwares técnicos na área de estimação de estado muitas vezes exige conhecimentos em programação ou leitura de documentação para inserção de dados e interpretação de resultados, o que pode dificultar sua aplicação prática. Nesse sentido, o presente trabalho apresenta o desenvolvimento de uma interface gráfica para simplificar a utilização de um software de estimação de estado harmônico aplicado ao sistema de distribuição IEEE de 34 barras. A interface foi desenvolvida com a biblioteca Tkinter, após a migração do código original, escrito em Matlab, para Python. Para validação, os resultados foram comparados com estudos prévios, utilizando os mesmos dados de entrada e verificando a consistência dos resultados de saída. Ainda, rotinas específicas foram criadas para auxiliar na migração, validação e depuração do código. Os resultados demonstraram equivalência total em relação ao software original, proporcionando uma interface intuitiva para inserção de dados, por meio de elementos gráficos interativos, e exibição simplificada dos resultados, facilitando sua análise e interpretação.

Palavras-chave—Interface Gráfica, Estimação de Estado Harmônico, Sistemas de distribuição, Linguagem Python, Qualidade da energia elétrica.

I. INTRODUÇÃO

A estimação do estado harmônico é uma metodologia usada em sistemas de energia para estimar os valores das tensões harmônicas e identificar fontes harmônicas. Tal metodologia envolve a seleção de variáveis de estado e o uso de variáveis de medição, como fasores de tensão de barra, fasores de corrente nas linhas e fasores de corrente injetada.

Vários métodos foram desenvolvidos para estimação do estado harmônico, incluindo aqueles baseados no processamento de sinais gráficos [1], aprendizado bayesiano esparso [2] e decomposição em valores singulares [3], [4]. Esses métodos visam realizar estimativas de estado sem calcular a impedância harmônica e podem obter resultados precisos mesmo com menos medições. No geral, a pesquisa

de estimação do estado harmônico se concentra em monitorar a qualidade da energia elétrica, estimando e identificando com precisão a “poluição harmônica” em sistemas de energia [5].

Contudo, um grande desafio para usuários de softwares de estimação de estado é compreender e lidar com os dados de entrada e os resultados obtidos. Isso exige um conhecimento aprofundado tanto da linguagem de programação utilizada quanto do código em si, dificultando sua utilização por pesquisadores e profissionais que não possuem esse domínio. Além disso, os resultados gerados podem ser extensos e complexos, tornando a análise visual mais trabalhosa e suscetível a erros.

Para resolver essa questão, este trabalho propõe o desenvolvimento de uma interface gráfica que simplifique a inserção de dados de entrada e a visualização dos resultados de saída de um estimador de estado harmônico aplicado a sistemas de distribuição de energia elétrica. Soluções semelhantes têm sido adotadas por outros autores, como [6] [7], [8], que identificaram a necessidade de criar interfaces gráficas para facilitar a utilização de seus softwares.

Neste trabalho, foi utilizado um código de estimador de estado harmônico originalmente desenvolvido por [3], que implementa um estimador baseado na decomposição em valores singulares, inicialmente escrito em Matlab. A migração para a linguagem Python foi essencial, dado o seu caráter gratuito, ampla utilização e suporte a bibliotecas que facilitam tanto o desenvolvimento de interfaces quanto a validação e depuração do código.

Assim, o objetivo principal deste trabalho é criar uma interface que torne o software mais acessível e intuitivo, permitindo uma interação eficiente e confiável, além de ampliar sua aplicação por pesquisadores e profissionais da área.

II. FUNDAMENTAÇÃO TEÓRICA

A. Estimação de estado harmônico

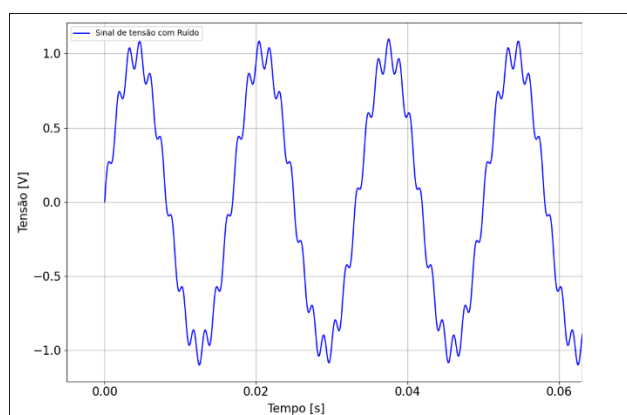
As concessionárias de energia possuem uma frequência fundamental em suas redes que chegam aos consumidores, sendo no Brasil de 60 Hz. Entretanto, isso pode não ocorrer em determinadas situações, já que alguns equipamentos e cargas com características não lineares acabam alterando essa frequência, fenômeno conhecido como distorção harmônica.

Os harmônicos são componentes de frequência que, idealmente, surgem em múltiplos inteiros da frequência fundamental e ocorrem de forma regular. Em um sistema de energia elétrica, quando esses harmônicos se combinam com as componentes fundamentais de corrente e tensão, eles criam distorções nas formas de onda [4]. A Fig. 1 apresenta um exemplo de distorção na forma de onda.

Nesse sentido, a estimação de estado harmônico é o monitoramento dessa distorção que influencia diretamente na qualidade de energia elétrica. Quando a qualidade é comprometida, os sistemas elétricos de potência podem sofrer diversos impactos negativos, como ([9], [10], [11], [12], [13], [14], [15], [16]):

- Sobreaquecimento;
- Mal funcionamento;
- Redução da vida útil de equipamentos eletrônicos;
- Possibilidade de ressonância.

Fig. 1. Distorção na forma de onda – Harmônico



B. O Uso do Método DVS Para Solução

A decomposição em valores singulares (DVS) é uma técnica matemática essencial para a manipulação de matrizes que não possuem inversa. Em comparação com outros métodos, como a eliminação de Gauss ou a decomposição LU, o método DVS oferece vantagens consideráveis, como a capacidade de identificar problemas em sistemas e propor soluções robustas, além de fornecer respostas numéricas mais precisas [3].

No contexto da estimação de estado harmônico (EEH), o método DVS se torna ainda mais relevante, pois ele possibilita a análise e a decomposição de sinais harmônicos em sistemas elétricos de forma eficiente. A capacidade do DVS de lidar com dados de alta dimensão, como os frequentemente encontrados em estudos de distorção harmônica, é fundamental para a precisão dos resultados obtidos. Quando aplicado a sistemas de distribuição de

energia elétrica, o DVS permite a estimativa precisa do estado harmônico, ajudando na identificação de distorções na forma de onda de tensão e corrente.

Uma grande propriedade do DVS em estimação de estado harmônico é sua capacidade não apenas de estimar as tensões nas barras de um sistema de distribuição, mas também de identificar locais onde a estimação não foi possível. Nesses casos, o DVS sugere possíveis locais para a instalação de medidores, permitindo que os responsáveis pelo sistema tomem ações corretivas e melhorem a qualidade da estimativa. Essa funcionalidade é destacada no trabalho de [3], que apresenta o método de estimação de estado harmônico utilizando DVS e enfatiza a importância dessa abordagem para a otimização do processo de monitoramento e a melhoria da precisão dos resultados. Assim, em [3], todo o equacionamento detalhado do método DVS aplicado à EEH pode ser encontrado, proporcionando uma metodologia para a estimação do estado harmônico no sistema de 34 barras do IEEE, entre outros.

C. Interfaces Gráficas via Python

No geral, uma interface gráfica de usuário é projetada para permitir que os usuários naveguem e manipulem o conteúdo por meio do toque, ou clique. Ela permite que os usuários interajam com itens de conteúdo discretos, como músicas, vídeos, documentos, páginas da web, e realizem ações como fazer uma chamada, ler uma mensagem ou jogar um jogo [17]. De forma resumida, a interface é uma ótima ferramenta que dispensa o entendimento complexos do código de programação de alguma aplicação para executá-la.

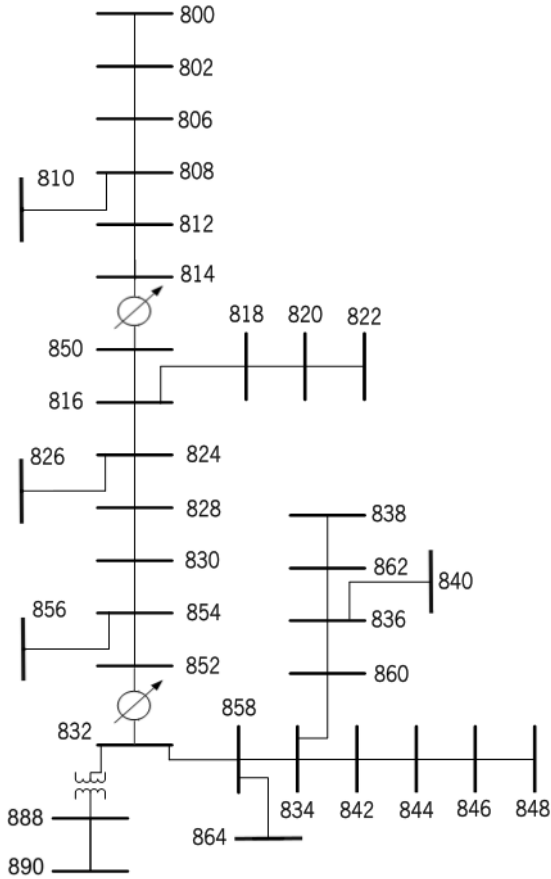
O uso de interfaces gráficas em softwares técnicos traz uma série de benefícios, principalmente no contexto de programas complexos como os utilizados para a estimação de estado harmônico. Interfaces gráficas permitem a redução da curva de aprendizado para usuários não familiarizados com programação, proporcionando uma interação mais intuitiva e acessível com o software. Substituindo comandos textuais por elementos visuais — tais como botões, campos de entrada e gráficos interativos — a interface gráfica não apenas facilita a inserção e análise de dados, mas também reduz a probabilidade de erros humanos, aumentando a eficiência do software [18] [19].

No desenvolvimento dessa interface, a linguagem de programação Python aparece como uma das opções. Python oferece uma gama de bibliotecas, como Tkinter, que facilitam a criação de interfaces gráficas intuitivas, além de permitir a integração com outras bibliotecas, como NumPy e Matplotlib, que são essenciais para a análise e visualização de dados. A flexibilidade de Python, aliada à sua comunidade ativa e ao fato de ser uma linguagem de código aberto, torna-a uma escolha adequada para pesquisadores e desenvolvedores que buscam soluções mais acessíveis e eficientes. Sua aplicabilidade se estende além da interface gráfica, sendo também uma ferramenta poderosa para a migração e análise de códigos complexos [20].

III. METODOLOGIA

O sistema de distribuição utilizado para as simulações foi o IEEE de 34 barras, ilustrado na Figura 2. Para alcançar os resultados propostos, foi necessário compreender os conceitos fundamentais relacionados à estimação de estado harmônico e ao código original em Matlab [3].

Fig. 2. Sistema de distribuição de 34 barras do IEEE.



No que diz respeito ao EEH, foram abordados aspectos chave como sistemas de distribuição, distorções harmônicas e o próprio processo de estimação do estado harmônico, os quais foram essenciais para entender o problema e buscar uma solução viável.

Quanto ao código em Matlab, foi fundamental entender o ambiente de desenvolvimento, bem como a correlação entre os dados de entrada e saída do programa e suas implicações práticas para os sistemas de distribuição de energia. Esse entendimento foi importante para a adaptação do código e sua migração para a linguagem Python.

A. Dados de Entrada e Saída

Nos próximos parágrafos, será relatado como se deu a compreensão dos dados de entrada e saída do algoritmo, com ênfase na análise desses dados para posterior conversão do código para Python.

Os dados de entrada do programa consistem em vetores binários que representam a presença ou ausência de medidores de tensão e corrente. Em cada vetor, um valor de 0 indica a ausência de um medidor e 1 indica sua presença. Cada posição do vetor corresponde a uma barra do sistema de 34 barras, com 34 medidores de tensão (um em cada

barra) e 66 de corrente (um para cada extremidade das 33 linhas do sistema), totalizando 100 medidores possíveis.

Para definir a posição dos medidores no sistema, é necessário compreender que cada índice nos vetores *med_tensao* e *med_corrente* corresponde a uma posição específica no sistema de 34 barras, conforme descrito na documentação do código.

Na Figura 3, as posições dos vetores *med_tensao* e *med_corrente*, que variam de 1 a 100, estão detalhadas. Como mencionado anteriormente, essas posições estão associadas às barras reais do sistema de 34 barras.

Fig. 3. Posição dos medidores nos vetores *med_tensao* e *med_corrente*

```
ned_tensao = [0 1 0 0 1 1 0 1 0 1 1 1 0 1 1 1 0 ... 33 34];
ned_corrente = [0 1 1 0 0 0 0 0 1 1 1 0 0 0 1 0 0 ... 98 99 100];
```

Para compreender o funcionamento do software desenvolvido em Matlab e os valores de entrada e saída, é necessário definir os parâmetros de entrada que são passados para a função EEH. Na Figura 4, é apresentado um exemplo de código que define os vetores de medidores, configurados aleatoriamente, e os passa para a função EEH, conforme os comentários atribuídos ao código.

Fig. 4. Exemplo de código chamando a função EEH em Matlab com um cenário aleatório.

```
%Definindo os Medidores de Tensão
med_tensao = [0 0 0 0 1 0 0 1 0 1 1 1 0 1 1 1 0 1 0
1 0 1 0 0 1 1 0 1 1 1 1 0 1];

%Definindo os Medidores de Corrente
med_corrente = [0 1 1 0 0 0 0 0 1 1 1 0 0 0 1 0 0 0 0
0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0
1 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 1 1 0 0];

%Chamando a Função EEH com os Medidores Como Parâmetro
EEH(med_tensao,med_corrente);
```

Após definir os vetores *med_tensao* e *med_corrente* e chamar a função EEH com esses parâmetros, o código é executado e retorna um vetor ERRO de tamanho 204, que representa a diferença entre a tensão estimada e a esperada para todas as barras. Esse vetor informa se a estimativa da tensão harmônica foi bem-sucedida em cada uma das 34 barras, mas a informação não é fornecida de forma direta. Para interpretá-lo, é necessário realizar uma análise detalhada, como a apresentada na Tabela 2, onde os primeiros 102 valores correspondem às partes reais das tensões e os outros 102 valores às partes imaginárias. Esses resultados devem ser agrupados em conjuntos de três, uma vez que se referem às três fases do sistema.

Essa estrutura torna a análise do vetor *ERRO* extremamente complexa e suscetível a erros, caso seja realizada manualmente. Como pode ser observado, sem um processo automatizado, essa tarefa torna-se bastante desafiadora e propensa a falhas, dada a grande quantidade de dados a ser processada e interpretada.

Para que a tensão harmônica de uma barra seja estimada corretamente, tanto a parte real quanto a parte imaginária do valor da tensão devem ser iguais a 0. Caso qualquer um desses valores seja diferente de 0, a estimativa do estado harmônico para essa barra não será considerada válida.

TABELA 1. FORMA DE EXIBIÇÃO DOS RESULTADOS DA ESTIMAÇÃO NO MATLAB

Essa forma de visualização da Tabela 1 torna o processo de verificar quais barras tiveram a tensão estimada e quais não tiveram mais suscetível a erros, pois a análise dependia de uma verificação manual de cada valor. Sendo que para chegar em quais barras tiveram as tensões harmônicas estimadas, seria necessário um grande trabalho manual, semelhante ao da Tabela 2.

TABELA 2. RESULTADO DA ETIMACÃO PARA AS 34 BARRAS

B. Desenvolvimento da Interface Gráfica

A interface gráfica foi desenvolvida utilizando a biblioteca Tkinter do Python, com a imagem do sistema de 34 barras do IEEE como plano de fundo. Sobre essa imagem, foram desenhados os medidores de tensão e de corrente, o que facilitou consideravelmente o desenvolvimento da interface.

Na página de inserção de dados (Figura 5), o usuário pode facilmente inserir os medidores de tensão e corrente clicando com o botão esquerdo do mouse sobre a posição desejada. Para remover um medidor, basta clicar com o botão direito. Após a inserção dos medidores, o usuário pode clicar no botão "Estimar" para executar o software e apresentar os resultados na página seguinte (Figura 6), onde as barras com as tensões estimadas são destacadas.

A Figura 6 apresenta os resultados da estimativa de tensão de forma mais simples e intuitiva. Como indicado na legenda, as barras que tiveram seus valores de tensão estimados são destacadas visualmente por diferentes cores (verde) em relação às que não tiveram (vermelho), facilitando a análise dos resultados. Ao observar a figura, é possível identificar que as barras 800, 802, 806, 808, 812 e 814 não tiveram seus valores de tensão estimados.

Fig. 5. Inserindo medidores do cenário aleatório.

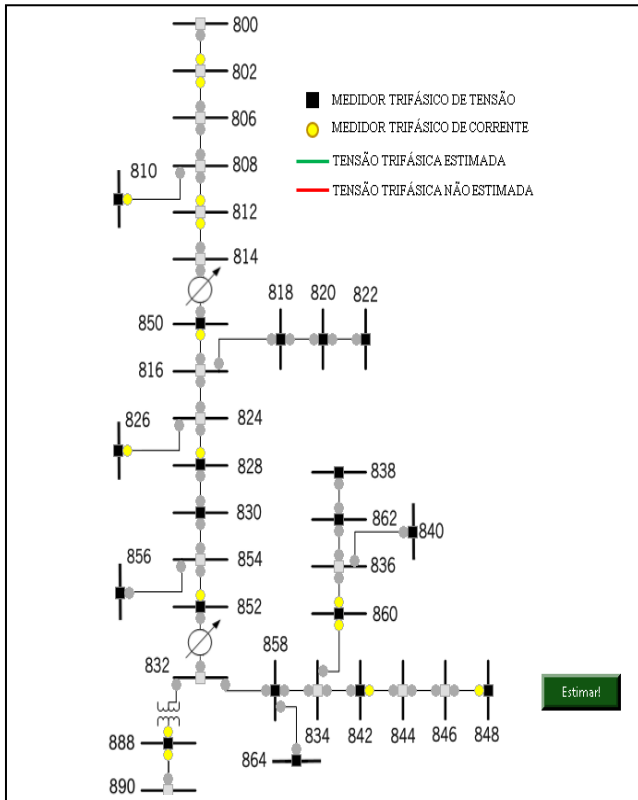
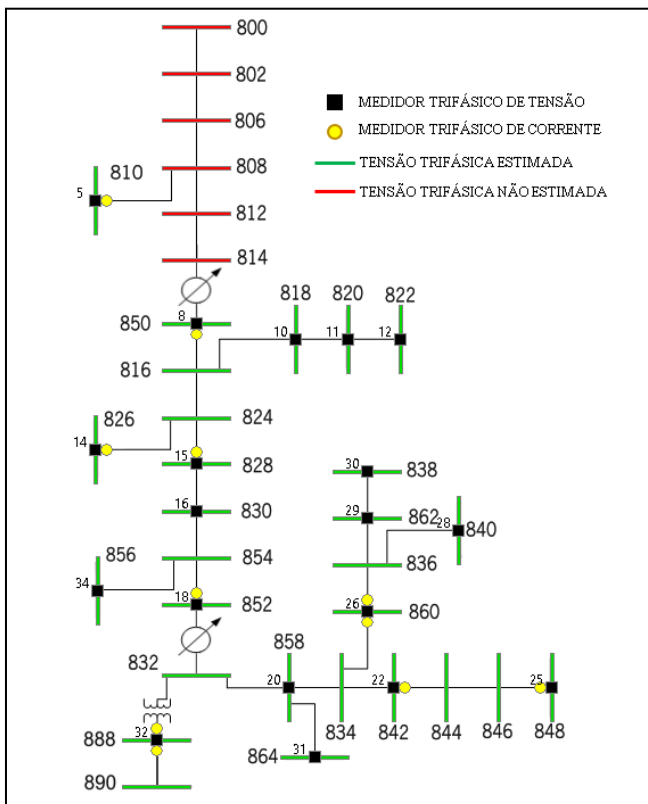


Fig. 6. Resultado da interface utilizando o cenário aleatório.



Com isso, para demonstrar que os resultados obtidos foram satisfatórios, a seção seguinte fará comparações com outros trabalhos, destacando as diferenças em relação à

configuração aleatória apresentada na Figura 4, que foi inicialmente utilizada apenas para ilustrar o funcionamento da interface.

IV. RESULTADOS

No intuito de demonstrar que os resultados foram satisfatórios observando além de uma configuração aleatória, a subseção seguinte demonstrará comparações deste trabalho com [4]. Para efeito de comparação, o trabalho [4] foi escolhido por utilizar o estimador original em Matlab, desenvolvido por [3], para obter os resultados apresentados no sistema de distribuição de 34 barras do IEEE levando em consideração situações reais de monitoramento, tais como:

- Alocação de medidores em barras onde ocorriam variações de tensão de curta duração em maior quantidade;
- Alocação dos medidores em barras que continham três ramificações no sistema de distribuição de 34 barras do IEEE;
- Alocação de medidores em barras que possuem cargas conectadas no sistema;
- Alocação de medidores em todas as barras nas extremidades do sistema;
- Cenário em que há a observabilidade total do sistema com 21 medidores de tensão alocados.

Neste trabalho foram comparados os resultados com dois dos cenários de [4], nos quais os medidores foram alocados em barras que continham cargas conectadas e o cenário em que foram alocados 21 medidores para observação total do sistema. Esses testes servem para validação do software através da comparação de resultados.

A. Cenário 1: Alocação em barras com cargas conectadas

No Cenário 1, foram alocados medidores de tensão nas barras 830, 840, 844, 848, 860 e 890. Os medidores de corrente foram posicionados no início de todas as linhas conectadas a essas barras. Os resultados apresentados no trabalho [4] estão mostrados na Figura 7, enquanto os resultados obtidos neste trabalho podem ser visualizados na Figura 8.

Ao comparar as Fig. 7 e 0, é possível observar que os resultados obtidos foram idênticos, estimando a tensão nas barras 828, 830, 834, 836, 840, 842, 844, 846, 848, 860, 888 e 890.

B. Cenário 2: Observabilidade completa do sistema

Além do Cenário 1, o Cenário 2, também analisado em [4], apresenta a observabilidade completa do sistema, por isso se torna interessante sua análise neste trabalho. Neste cenário, os medidores de tensão foram alocados nas barras 802, 810, 812, 850, 818, 820, 822, 826, 828, 830, 852, 858, 842, 848, 860, 840, 862, 838, 864, 888 e 856. Já os medidores de corrente foram posicionados nos ramais 802-800, 802-806, 810-808, 812-814, 826-824, 828-824, 888-832, 852-854, 850-816, 888-890, 860-834, 860-836, 842-844 e 848-846.

Fig. 7. Cenário 1 retirado do trabalho [4].

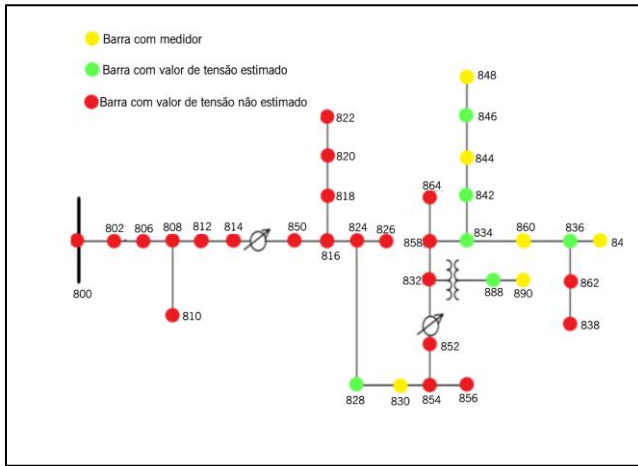
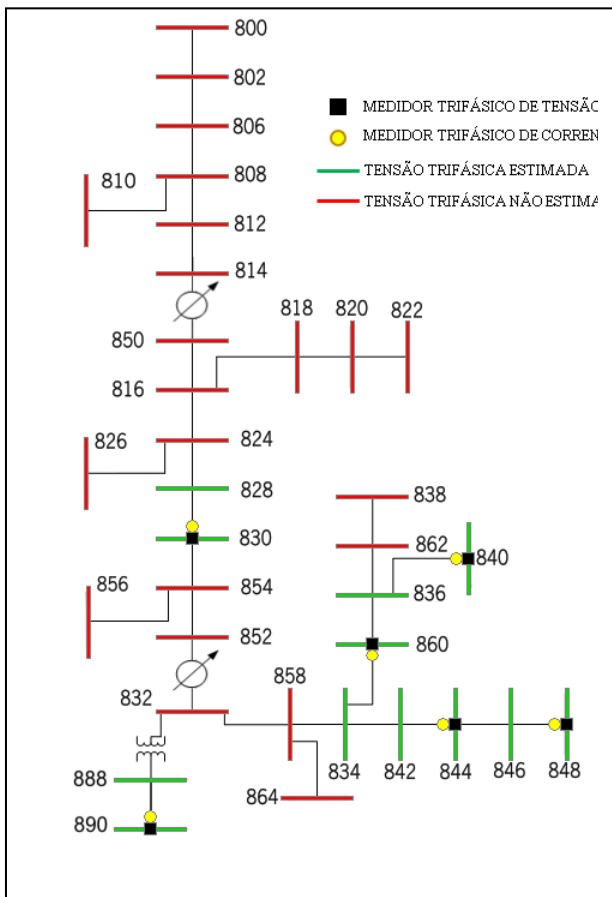


Fig. 8. Cenário 1 obtido pela interface gráfica.



Fazendo uma análise dos resultados de ambos os trabalhos no cenário 2 (Figuras 9 e 10), é possível observar que todas as barras tiveram suas tensões harmônicas estimadas, o que indica que o código transcrito em Python está validado, tendo obtido idênticos resultados nos dois cenários analisados.

É válido destacar que, nas Figuras 8 e Fig. 9 retiradas de [4], as barras onde estão alocados medidores de tensão são representadas por círculos de cor amarela, indicando que esses valores de tensão também estão sendo estimados. Além disso, tanto o método de Estimador de Estado Harmônico utilizado neste trabalho quanto o abordado em

[4] não consideram barras que possuam exclusivamente medidores de corrente alocados. Isso ocorre porque os medidores de corrente e de tensão fazem parte do mesmo equipamento, sendo incoerente uma configuração que contemple apenas medidores de corrente.

Fig. 9. Cenário 2 retirado do trabalho [4].

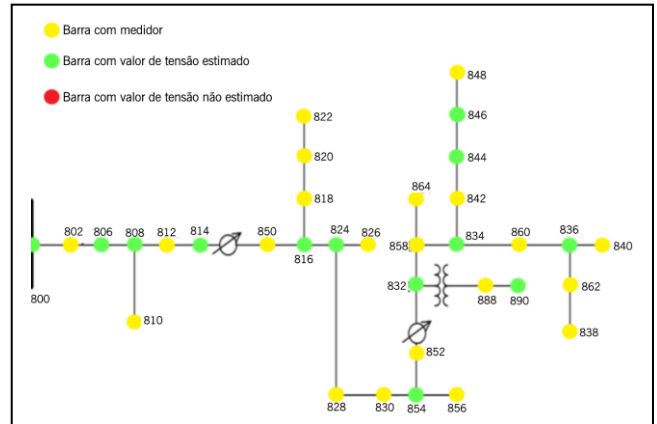
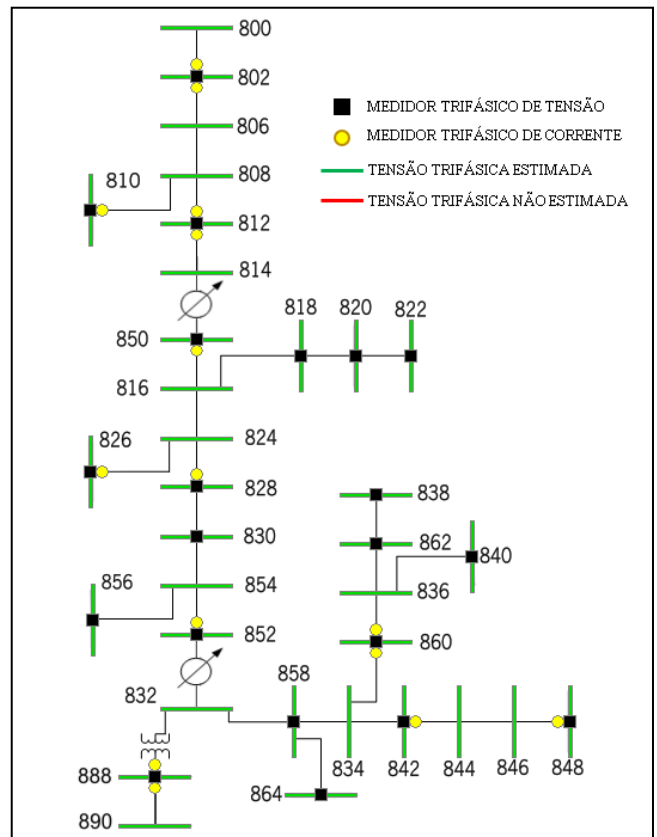


Fig. 10. Cenário 1 obtido pela interface gráfica.



V. CONCLUSÃO

No presente trabalho, uma interface gráfica para estimação de estado harmônico para sistemas de distribuição de energia elétrica foi desenvolvida utilizando a biblioteca Tkinter, com o objetivo de facilitar a visualização dos dados de entrada e saída, permitindo que até usuários não familiarizados com programação possam utilizá-la.

Para validar o código, foram criadas rotinas de comparação com resultados anteriores, o que assegurou a equivalência dos resultados entre o código Python e o software original, em Matlab. Dois cenários do trabalho [4] foram reproduzidos, e os resultados obtidos foram consistentes, o que confirma a precisão e confiabilidade da interface desenvolvida.

A criação da interface gráfica se mostrou crucial, uma vez que, anteriormente, o processo de inserção e análise de dados era suscetível a erros e exigia um trabalho considerável. A interface não só simplifica o processo, mas também melhora a precisão da análise ao eliminar erros visuais durante a verificação dos resultados.

Como perspectiva para trabalhos futuros, é necessário ampliar a funcionalidade da interface para outros sistemas de distribuição além do IEEE de 34 barras, permitindo que o usuário configure sistemas personalizados, seja por meio de arquivos texto ou pela interface, e o software possa, até mesmo, desenhar automaticamente o sistema.

VI. REFERÊNCIAS

- [1] Y. Wang, R. Xu, X. Ma, W. Qiu, Q. Fu e R. Zhou, "Harmonic state estimation based on graph signal processing.," *IET Generation, Transmission & Distribution*, vol. 16, nº 6, pp. 1083--1095, 2022.
- [2] Y. Yuan, W. Zhou, H.-T. Zhang, Z. Ping e O. Ardakanian, "Sparse Bayesian harmonic state estimation," *2018 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*, vol. 2018, nº IEEE, pp. 1-6, 2018.
- [3] J. Breda, "Sistema de monitoramento para estimação de estado harmônico trifásico para sistemas de distribuição utilizando decomposição em valores singulares," *Universidade de São Paulo*, 2017.
- [4] J. F. Oliveira, J. V. Gomes de Araújo, J. F. Dias Breda, J. C. de Melo Vieira Júnior, e J. F. Oliveira, "Estimação de Estado Harmônico Para Sistemas de Distribuição de Energia Elétrica Frente a Condições Reais de Monitoramento," *Simpósio Brasileiro de Sistemas Elétricos-SBSE*, vol. 1, p. 1, 2020.
- [5] Y. Zhang, Y. Xu e Y. Xu, "Research on power system harmonic state estimation," *2011 4th International Conference on Electric Utility Deregulation and Restructuring and Power Technologies (DRPT)*, nº 4, pp. 596--600, 2011.
- [6] L. MacFarland e G. J. Streletz, "Using Python to Develop Graphical Interfaces to Scientific Data," *Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States)*, 1999.
- [7] I. G. Y. Rabello, M. A. Jucá e F. da Rocha Henriques, "Desenvolvimento de uma Interface Gráfica para Simulador de Meios ópticos Guiados," *XL Simpósio Brasileiro de Telecomunicações e Processamento de Sinais*, 25--28 Setembro 2022.
- [8] H. R. Mendes, R. T. Massera e A. Tomal, "pyPEN: uma interface gráfica user-friendly para simulação Monte Carlo em física médica," *Revista Brasileira de Ensino de Física*, vol. 44, p. 06, 2022.
- [9] E. W. KIMBARK, *Direct Current Transmission*, Wiley-Blackwell; Volume 1 edição, 1971.
- [10] E. F. Fuchs, D. J. Roesler e F. S. Alashhab, "Sensitivity of electrical appliances to harmonics and fractional harmonics of the power system's voltage. Part I: Transformers and induction machines," *IEEE Transactions on Power Delivery*, vol. 2, nº 2, pp. 437-444, 2007.
- [11] E. F. Fuchs, D. J. Roesler e K. P. Kovacs, "Sensitivity of electrical appliances to harmonics and fractional harmonics of the power system's voltage. Part II: Television sets, induction watt-hour meters and universal machines," *IEEE Transactions on Power Delivery*, vol. 2, nº 2, pp. 445-453, 1987.
- [12] IEEE, "IEEE recommended practices and requirements for harmonic control in electrical power systems," *New York, NY, USA*, pp. 1-1, 1993.
- [13] IEEE, "Practical definitions for powers in systems with nonsinusoidal waveforms and unbalanced loads: a discussion," *IEEE Transactions on Power Delivery*, vol. 11, nº 1, pp. 79-101, 1996.
- [14] J. C. Das, *Power system analysis: short-circuit load flow and harmonics*, CRC press, 2002.
- [15] G. J. Wakileh, *Power systems harmonics: fundamentals, analysis and filter design*, Springer Nature, 2019.
- [16] H. Wang, "Harmonic impact of modern residential loads on distribution power system and mitigation solutions," *University of Alberta*, 2011.
- [17] W. D. B. Vaughan e D. E. Vaughan, *Graphical user interface*, 2016.
- [18] F. P. Oliveira, "Interfaces Gráficas para Softwares Técnicos: Benefícios e Aplicações," *Journal of Computational Engineering*, 2015.
- [19] R. S. Silva e M. Costa, "Interface Gráfica para Estudo de Sistemas Elétricos de Potência: Uma Abordagem em Python.," *IEEE Transactions on Power Systems*, 2018.
- [20] J. & L. S. Beasley, *Python for Scientific Computing: Applications in Engineering and Mathematics*, Springer, 2020.