

Capítulo 1

Exercícios

Diferente da lista de exercícios anterior, que tinha como objetivo desenvolver o raciocínio lógico e familiarizar o leitor com os mecanismos básicos da linguagem Python, esta nova lista propõe desafios mais complexos, voltados para situações comuns no cotidiano profissional de engenheiros.

O foco agora é estimular a capacidade de resolver problemas e de traduzi-los para o ambiente computacional, utilizando o Python como ferramenta de apoio.

Questão 1.1: [pll](#)

Em sistemas de geração fotovoltaicos a energia absorvida pelos arranjos de painéis é convertida em energia elétrica em corrente contínua. Como na maioria dos sistemas não há nenhuma forma de armazenamento, e como energia se conserva, toda energia absorvida pelos módulos deve ser prontamente injetada na rede. Para escoar a produção, os inversores de frequência devem não apenas executar a conversão CC-CA, mas fazê-la de forma a controlar a injeção de potência na rede, garantindo o equilíbrio energético e consequentemente a estabilidade do barramento CC.

Para tal, a tensão imposta pelo inversor deve estar em sincronismos com a da rede. De fato, o controle da potência ativa do inversor depende da fase relativa entre a tensão da rede e a tensão imposta pelo inversor. Ainda, como a rede opera com frequência variável, em torno dos 60 Hz, o inversor precisa a todo instante estimar a fase da tensão da rede para executar os ajustes necessários de modo a manter o sincronismo e o ângulo de carga desejado.

Essa estimação não é trivial já que, além das variações da tensão e frequência em regime permanente, a rede ainda apresenta efeitos transitórios impulsivos e oscilatórios, distorções harmônicas, flutuações, ruídos, afundamentos, etc. Isso é ainda mais desafiador em sistemas monofásicos, onde amplitude, frequência e fase devem ser obtidas a partir de medições de um único sinal sujeito aos fenômenos já citados.

Na prática, existem algumas estratégias para se obter uma estimativa instantânea e precisa da fase da tensão da rede, sendo que a maioria delas são baseadas na estrutura SOGI-PLL, cujo diagrama de blocos básico é apresentado na Figura 1.1.

O estimador da Figura 1.1 pode ser discretizado e reescrito na forma de equações a diferenças,

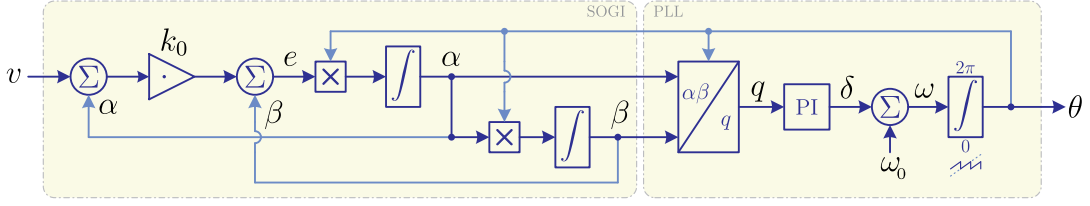


Figura 1.1: SOGI-PLL.

apresentadas nas equações (1.1–1.7), permitindo sua implementação em controladores digitais.

O algoritmo discretizado recebe um sinal amostrado da tensão de entrada (geralmente em algumas poucas dezenas de kHz, dependendo da frequência de comutação empregado no inversor). A partir dessas amostras, calcula a fase quase instantânea (com atraso de uma amostra). Esse sinal é utilizado nas malhas de controle das correntes e da potência injetadas na rede (não apresentados).

$$e_n = k_0 (v_n - \alpha_{n-1}) - \beta_{n-1} \quad (1.1)$$

$$\alpha_n = w_{n-1} k_1 (e_n + e_{n-1}) + \alpha_{n-1} \quad (1.2)$$

$$\beta_n = w_{n-1} k_1 (\alpha_n + \alpha_{n-1}) + \beta_{n-1} \quad (1.3)$$

$$d_n = \alpha_n \sin(\theta_{n-1}) - \beta_n \cos(\theta_{n-1}) \quad (1.4)$$

$$\delta_n = k_2 q_n + k_3 q_{n-1} + \delta_{n-1} \quad (1.5)$$

$$\omega_n = \omega_0 + \delta_n \quad (1.6)$$

$$\theta_n = k_1 (\omega_n + \omega_{n-1}) + \theta_{n-1} \pmod{2\pi} \quad (1.7)$$

Em <https://github.com/j-Lago/NVP68300> você encontrará um arquivo **rede.csv** com a tensão (coluna 2) amostrada de alguns ciclos da rede elétrica e um registro temporal (coluna 1) de cada amostra. Escreva um script que importe esse sinal para um vetor, e para cada amostra (cada elemento do vetor) execute os cálculos (1.1–1.7).

Guarde os resultados de cada amostra de v (tensão de entrada), α (tensão filtrada), d (componente de eixo direto da tensão em base síncrona), q (componente em quadratura) e θ (fase estimada pelo PLL). Gere e exporte uma figura apresentando a evolução temporal de cada uma desses sinais, bem como alguma indicação do instante que o PLL conseguiu se sincronizar (atracar) com a rede elétrica. De forma simplificada, assumo que o sincronismo foi obtido no instante do primeiro cruzamento por zero do sinal θ (saída do PLL) em que $\arctan(q/d) < 0.035 \text{ rad} (\approx 2^\circ)$ e $d > 250 \text{ V}$. A Figura 1.2 mostra um exemplo de como os resultados do script podem ser apresentados.


 **../figs/plot2.pdf**

Figura 1.2: Resultado esperado do script.

time [s]	PLL angle [rad]	grid voltage [V]
0.12562	, 0.00158479	, 4.3073
0.12562	, 0.00370203	, 14.323
0.12563	, 0.005819309	, -0.32799
0.12564	, 0.007936621	, 14.62
0.12564	, 0.010053976	, 14.767
⋮	⋮	⋮

Tabela 1.1: Exemplo da tabela a ser exportada em formato `.csv`.

Assuma as seguintes constantes para a implementação do controlador e das integrações:

$$\begin{aligned}
 k_0 &= 0.7 \\
 k_1 &= 2.840000000003951 \cdot 10^{-6} \\
 k_2 &= 0.385715277950311 \\
 k_3 &= -0.385713293478261 \\
 \omega_0 &= 376.9911184307752
 \end{aligned}
 \tag{1.8}$$

Ainda, exporte para um arquivo `.csv` a porção do sinal de tensão de entrada compreendida entre a detecção da sincronização até o último cruzamento por zero do ângulo θ (período que compreende todos os ciclos inteiros em que o PLL estava sincronizado). Exporte além da tensão, o período correspondente do vetor de tempo e do ângulo de saída do PLL.