Relatório da simulação do conversor Boost- Ana Gabriela.

O conversor Boost é um conversor elevador de tensão, com princípio de funcionamento semelhante ao conversor Buck, com a diferença que agora os componentes serão reposicionados visando obter uma operação diferente daquela do converso abaixador de tensão. Neste relatório será analisada a simulação do conversor boost apresentada na aula de eletrônica de potencia do curso de engenharia elétrica da UNEMAT de Sinop.

O código será separado e 9 passos que serão descritos abaixo:

1° Passo foi a importação das bibliotecas utilizadas para as operações matemáticas e plotagem do gráfico:

```
1   import numpy as np
2   import matplotlib.pyplot as plt
3   from scipy.integrate import solve_ivp
4
```

- numpy → usada para cálculos numéricos e manipulação de vetores (np.linspace, np.zeros_like, etc).
- matplotlib.pyplot → usada para plotar gráficos de tensão e corrente.
- solve_ivp → função da biblioteca scipy que resolve equações diferenciais ordinárias (ODEs) numericamente.

2° passo é a definição dos parâmetros do circuito:

```
5   # ------------------------------------
6   # Parâmetros do conversor Boost
7   # ------------------------------------
8   Vin = 12.0
9   Vout = 24.0
10  Pout = 10.0
11  fs = 100e3
12  Ts = 1/fs
13  D = (Vout - Vin)/Vout    # Ciclo de trabalho
14
```

- Vin = tensão de entrada do conversor (V).
- Vout = tensão de saída desejada (V).
- Pout = potência do sistema (W).
- fs = frequência de chaveamento do conversor (Hz).
- Ts = período de chaveamento: $Ts = 1/fs$.
- D = ciclo de trabalho do boost: $D = \frac{Vout - Vin}{Vout}$. Ele indica a fração do tempo em que a chave fica **ligada**.

3° Passo a definição das Ondulações de Corrente e Tensão:

```
15  # Ondulações desejadas
16  delta_IL = 0.10        # 10% da corrente média
17  delta_VC = 0.01        # 1% da tensão
18
```

As ondulações desejadas (ripple) em um conversor, como o boost, são valores-alvo que definimos para a variação máxima aceitável da corrente do indutor e da tensão do capacitor. Elas determinam quanto a corrente e a tensão podem "oscilar" em torno de seus valores médios durante o funcionamento do conversor.

Para a corrente temos:

$$\Delta IL = \text{percentual desejado} \times \text{Imédia}$$

- Se a ondulação desejada for muito pequena o indutor é maior, se for muito grande o indutor é menor, mas maior ripple e mais perda.
- queremos que a corrente do indutor oscile no máximo 10% da corrente média de entrada.
- Sera usada para calcular o indutor mínimo necessário.

Para a tensão temos:

$$\Delta VC = \text{percentual desejado} \times \text{Vout}(\text{tensão média de saída})$$

- queremos **1% de ripple na tensão de saída**.
- Serve para calcular o **capacitor mínimo necessário**.

- Ripple pequeno= capacitor maior, saída mais estável.
- Ripple grande= capacitor menor, saída mais instável.

4° Passo definição de Iin, Iout e R:

```
19    # Cálculos auxiliares
20    Iin = Pout / Vin
21    Iout = Pout / Vout
22    R = Vout**2 / Pout
23
```

- Iin= corrente média de entrada do conversor.
- Iout= corrente média de saída.
- R= resistência equivalente da carga

5° Passo dimensionamento do indutor e capacitor:

```
24    # Dimensionamento
25    L = Vin * D / (fs * delta_IL * Iin)
26    C = Iout * D / (fs * delta_VC * Vout)
27
```

Fórmulas clássicas para **boost em modo contínuo**:

$$L = \frac{V_{\text{in}} \cdot D}{f_S \cdot \Delta I_L \cdot I_{\text{in}}} \quad \text{e} \quad C = \frac{I_{\text{out}} \cdot D}{f_S \cdot \Delta V_C \cdot V_{\text{out}}}$$

- L= indutor necessário para manter a ondulação de corrente dentro de 10%.
- C= capacitor necessário para manter a ondulação de tensão dentro de 1%.

6 ° Passo Definição da função do conversor:

```
28  # -------------------------------------
29  # Dinâmica do Boost
30  # -------------------------------------
31  def boost_ode(t, x):
32      iL, vC = x
33
34      if (t % Ts) < D * Ts:
35          vL = Vin
36      else:
37          vL = Vin - vC
38
39      di = vL / L
40      dv = (iL - vC/R) / C
41      return [di, dv]
42
```

- boost_ode= função que representa as equações diferenciais do circuito.
    - t= tempo atual da simulação.
    - x= vetor de estados do sistema
- (t % Ts) < D*Ts → determina se a chave está ligada (ON) ou desligada (OFF).
    - **(t % Tsw)=** devolve quanto tempo se passou desde o início do ciclo atual, sempre "reseta" para zero a cada período de chaveamento.
    - **D*Tsw**= compara se estamos na fase ON do PWM:
        - D*Tsw = tempo que a chave fica ligada (ON)
        - Se (t % Tsw) < D*Tsw → estamos no ON
        - Caso contrário → OFF
- ON: vL = Vin
- OFF: vL = Vin - vC
- di = vL / L = derivada da corrente do indutor.
- dv = (iL - vC/R) / C = derivada da tensão no capacitor.
- return [di, dv] = retorna a taxa de variação, que solve_ivp integra numericamente.

7° Passo defini a simulação:

```
43  # ------------------------------------
44  # Simulação
45  # ------------------------------------
46  t_end = 10e-3
47  t_eval = np.linspace(0, t_end, 200000)
48
49  x0 = [0.0, Vin]
50
51  sol = solve_ivp(
52      boost_ode,
53      (0, t_end),
54      x0,
55      method='RK45',
56      t_eval=t_eval,
57      max_step=Ts/50
58  )
59
```

- t_end= tempo final de simulação, 10 ms.
- t_eval= vetor de pontos onde queremos **avaliar** a solução.
- x0 = Condições iniciais (corrente inicial do indutor=0 e tensão inicial do capacitor começa igual à entrada)

**Integração com solve_ivp:**

- solve_ivp resolve as equações diferenciais.
- method='RK45' é o método Runge-Kutta adaptativo de 4ª ordem.
- t_eval= pontos para salvar o resultado.
- max_step=Ts/50  força a integradora a usar passos pequenos para capturar cada ciclo do conversor.
    - `Ts` é o período de chaveamento (ex: 1/100 kHz = 10 µs)
    - `Ts/50` obriga o integrador a nunca avançar mais do que um passo de 10 µs / 50 = 200 ns.

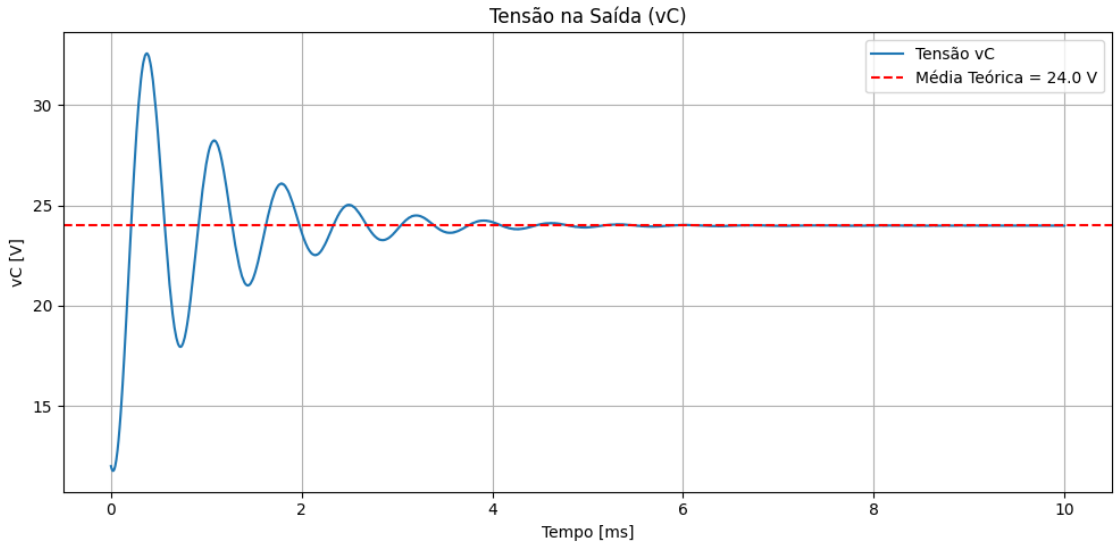
Esse é um ponto MUITO importante porque em conversores chaveados as dinâmicas mudam muito rápido, a chave liga e desliga com PWM, a corrente precisa ser calculada com alta resolução. Se o passo fosse grande demais o integrador iria “pular” a mudança ON/OFF, o modelo ficaria impreciso, apareceriam erros na forma de onda, as vezes a simulação até diverge.

Então max_step=Ts/50 garante uma simulação bem detalhada e estável.

8° Passo é a plotagem do gráfico da tensão.

```
# ------------------------------------
# Plotagem da tensão
# ------------------------------------
plt.figure(figsize=(10,5))
plt.plot(sol.t*1e3, sol.y[1], label='Tensão vC')
plt.axhline(Vout, color='red', linestyle='--', label=f'Média Teórica = {Vout} V')
plt.title("Tensão na Saída (vC)")
plt.xlabel("Tempo [ms]")
plt.ylabel("vC [V]")
plt.grid(True)
plt.legend()
plt.tight_layout()
plt.show()
```

9° Último passo geração e análise gráfica da tensão de saída.



O gráfico mostra a tensão de saída do conversor Boost ao longo do tempo durante a simulação, desde quando o sistema é energizado até alcançar o regime permanente.

A tensão inicia em torno de **12 V**, pois o capacitor começa carregado com a tensão de entrada definida, logo após a chave começar a operar, a tensão sobe rapidamente acima do valor desejado.

Nos primeiros milissegundos, há sobretensão e oscilações (overshoot + ripple transitório), isso ocorre porque indutor e capacitor formam um sistema de 2ª ordem, sujeito a oscilações quando energizado subitamente, a energia ainda está se redistribuindo entre os componentes do circuito.

Após o transitório, a tensão converge para aproximadamente 24 V, linha tracejada em vermelho. As oscilações vão diminuindo, por fim sistema estabiliza.