UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
CAMPUS DE SÃO JOÃO DA BOA VISTA

TUTORIAL TÉCNICO DA IMPLEMENTAÇÃO NUMÉRICA DO MÉTODO *BLADE ELEMENT MOMENTUM*

DISCIPLINA: AERODINÂMICA DE HELICÓPTEROS
PROF. DR. CARLOS DO CARMO PAGANI JUNIOR

ARNALDO P. STECCA FILHO - RA: 162491001
MARCY W. S. BORGES - RA: 162490917
RENATO MATOS DE OLIVEIRA - RA: 162490861
SÉRVIO TÚLIO S. H. BASTOS - RA: 162491018

MARÇO/2021

Sumário

1 Introdução

O método BEM (*Blade Element Momentum*) é empregado para o cálculo de parâmetros aerodinâmicos e de desempenhos de rotores de helicópteros, é baseado na combinação das teorias da quantidade de movimento, também conhecida como teoria do disco atuador, e da teoria do elemento de pá. De acordo com (SEDDON, 2011) , a primeira modela o rotor com um número infinito de pás de espessura desprezível, por outro lado, a segunda assume que a pá é composta por elementos bidimensionais analisados individualmente, sendo que cada elemento possui características geométricas e aerodinâmicas específicas.

A implementação do método avalia numericamente as distribuições radiais das seguintes quantidades: velocidade induzida adimensional (λ_i), gradiente do coeficiente de tração (dC_T/dr), coeficiente de sustentação (C_l), gradiente do coeficiente de torque (dC_Q/dr), fator de potência induzida (k), coeficiente de potência total (C_P) e figura de mérito (FM) foi implementada utilizando a metodologia apresentada por (SEDDON, 2011) e (LEISHMAN, 2006).

A implementação numérica deste tutorial apresenta a opção de escolha da inclusão de correção de ponta de pá, utilizando o modelo de Prandtl e, também, a inserção ou não de um modelo do arrasto de perfil, utilizando um valor constante ou um modelo da polar de arrasto parabólica. Além disso, é possível considerar afilamento na corda da pá. Por fim, o modelo implementado pode considerar torção ideal ou também uma distribuição linear do ângulo de torção.

Quanto às limitações, o modelo não apresenta correções de compressibilidade do escoamento, também não prevê casos de sustentação negativa, bem como a interferência das pás no escoamento.

2 Implementação numérica

A implementação foi executada usando da técnica de Programação Orientada a Objeto (POO). O pacote numérico “aerobemt” é composto por duas classes onde, na primeira, são definidas as características do rotor, sendo nomeada como “Rotor” e, na segunda, são implementados os métodos (funções) que compõem o *solver*, essa classe é denominada “Bemt”.

2.1 Classe “Rotor”

Nessa classe são definidos os parâmetros do rotor em que deseja-se realizar a simulação.

Os parâmetros obrigatórios de entrada são:

- solidEqui: Solidez equivalente do rotor;

- numberBlades: Número de pás do rotor;
- cla: Inclinação da curva de sustentação em 1/rad;
- ctreq: Coeficiente de tração requerido, esse parâmetro pode ser uma lista de valores;
- twinst: Distribuição de torção em grau.

Entretanto, há parâmetros que são opcionais e seus valores padrões (*default*) são apresentados a seguir:

- gamma: É a razão de afilamento da pá e, por *default*, gamma é igual um, que representa uma pá retangular;
- offset: Representa o tamanho da região de *cut-out* em que, por *default*, é igual a 10% do raio adimensional da pá;
- cd0: Parâmetro do arrasto constante, por *default* é igual a zero;
- d1: Parâmetro do arrasto que varia linearmente com AoA, por *default* é igual a zero;
- d2: Parâmetro do arrasto que varia parabolicamente com AoA, por *default* é igual a zero.

2.2 Classe "Bemt"

Classe onde são definidas as características da simulação e onde são implementados os métodos (funções) para o solver.

Os parâmetros obrigatórios desta classe são:

- Rotor: É uma variável tipo classe que representa os parâmetros definidos na Classe Rotor.

Parâmetros opcionais:

- Correção: Variável booleana que insere correção de ponta de pá no modelo (método de Prandtl);
- TwistIdeal: Variável booleana que insere a condição de torção ideal;
- elementosPa: Representa o número de elementos que compõem a discretização da pá, por *default* este parâmetro é igual a 100.

2.2.1 Métodos (função):

Dentro da classe "Bemt" existem vários métodos (funções) que realizam os cálculos de desempenho do rotor, podemos destacar duas dessas funções, a primeira é uma função de acesso interno da classe "Bemt" ("_BEMT") responsável por realizar a convergência de velocidade induzida e do ângulo de torção inicial, utilizando o método numérico apresentado no fluxograma da Figura 1, nesta função faz-se a inclusão ou não da correção de ponta de pá através do método de Prandtl. A segunda, é a função "*solver*" que é responsável por calcular os parâmetros de desempenho utilizando a velocidade induzida calculada em "_BEMT". As demais funções que compõem a classe "Bemt" são simples implementações de equações que são empregadas na função "*solver*", essas equações podem ser encontradas na literatura, ver por exemplo Seddon (2011) e Leishman (2006). As equações do método numérico empregado para a convergência da velocidade induzida e do ângulo de torção inicial e para o cálculo das funções empregadas no "*solver*" estão descritas no apêndice deste documento.

Para valores de C_T muito pequenos e considerando distribuição de torção linear, o método BEMT não converge na região da ponta da pá, uma vez que, como destacado nas limitações da metodologia empregada, este método não inclui a previsão de coeficiente de sustentação negativo. Sendo assim, a função implementada retorna um valor igual a zero nos pontos em que a teoria não converge. Ademais, na plotagem das curvas que incluem parâmetros limitantes da teoria, deve-se adicionar uma máscara a fim de eliminar os pontos nulos do vetor.

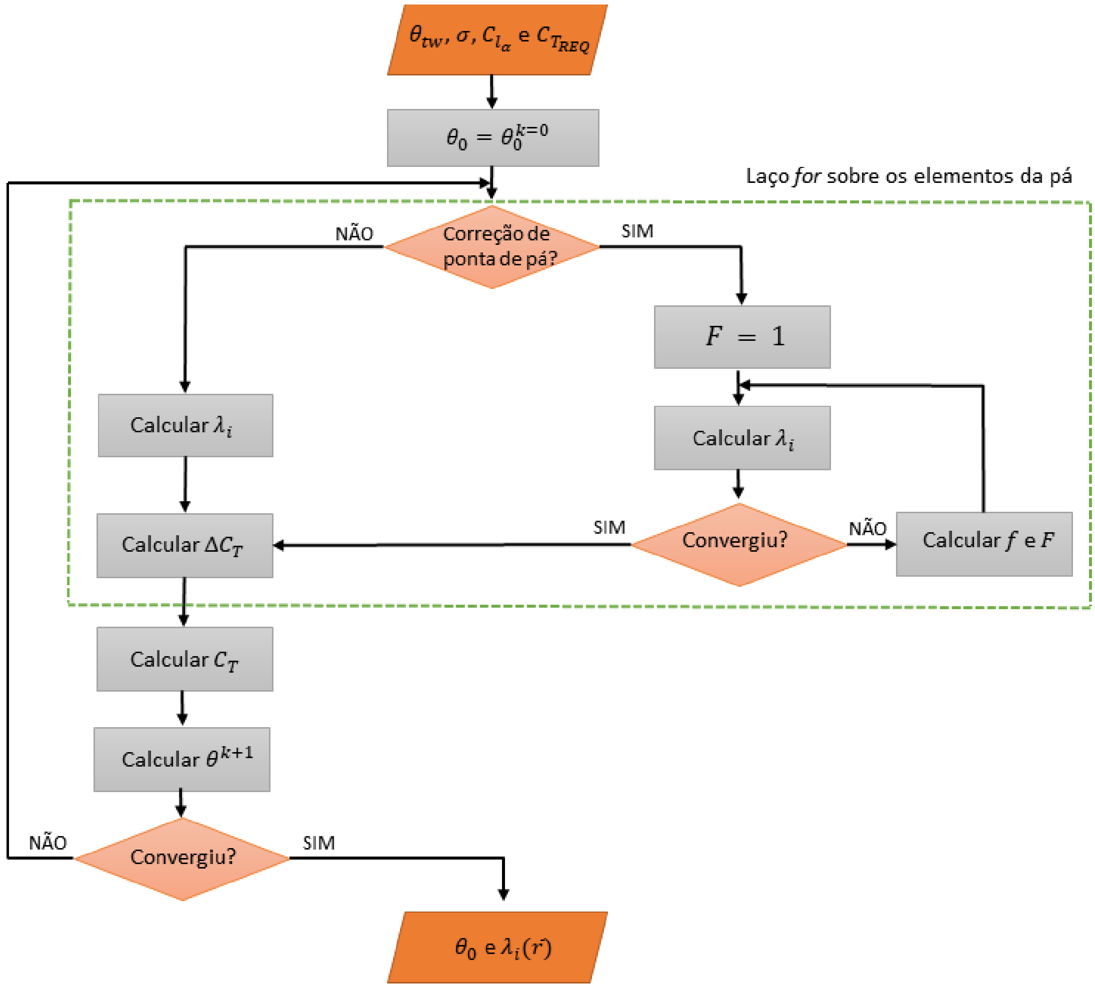


Figura 1: Fluxograma para a obtenção da velocidade induzida (λ_i) e do valor de θ_0.

A Figura 2 apresenta a divisão do pacote nume´rico nas duas classes consideradas e a relação entre elas. Vê-se que, como apresentado anteriormente, os elementos definidos na classe “Rotor” são *inputs* na classe “Bemt”, além disso, para obter os parâmetros de desempenho (*outputs*) deve-se utilizar a função *solver* que compõe a classe “Bemt”.

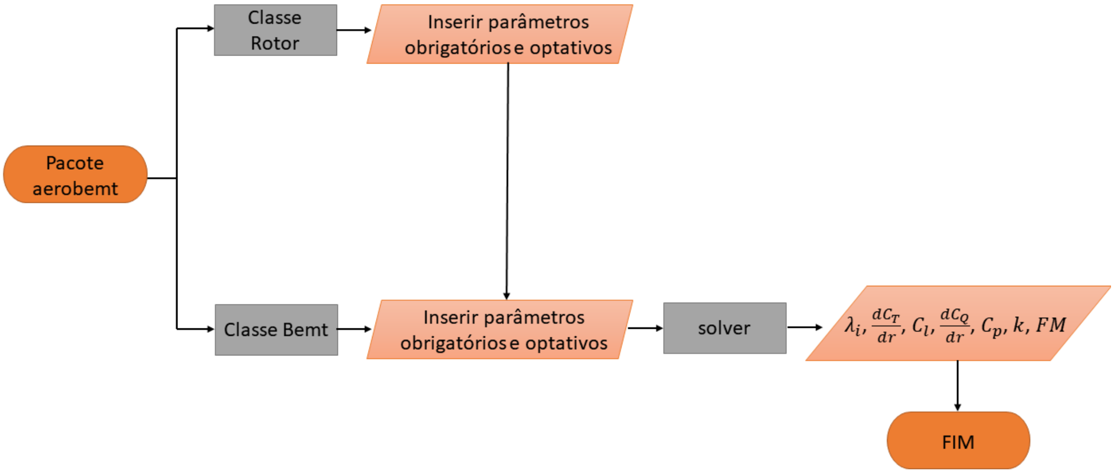


Figura 2: Fluxograma da divisão do pacote numérico em duas classes e a relação entre elas.

3 Tutoriais

A seguir será apresentado um tutorial introdutório de utilização do pacote “aerobemt”. Para esse tutorial considera-se os seguintes dados do rotor Dados: $\sigma = 0.1$, C_{treq} = 0.008, $C_{l\alpha}$ = 5.9, θ_{tw} = -10°, N_b = 2, C_{D0} = 0.01, d_1 = 0.025, d_2 = 0.65 e γ = 2. Considerando ainda correção de ponta de pá e uma discretização de 200 elementos.

- **Passo 1:** Inicialmente, importamos a biblioteca do matplotlib, responsável pelo *plots* e a biblioteca do “aerobemt”.

```
import matplotlib.pyplot as plt
# importando o modulo implementado
import aerobemt as bemt
```

- **Passo 2:** Em seguida, precisamos definir o rotor e seus parâmetros para isso, usamos a Classe “Rotor”, introduzindo os parâmetros obrigatórios e os parâmetros opcionais.

```python
# definindo um rotor


rotorTeste1 = bemt.Rotor(cla = 5.9, ctreq = [0.008] , solidEqui = 0.1,
                         numberBlades = 2, twist = -10, offset = 0.1)

# definindo parametros extras (não obrigatórios)
## parametro de afilamento
rotorTeste1.setAfilamento( gamma=2 )
## parametro de arrasto
rotorTeste1.setArrato( cd0 = 0.01, d1 = 0.025, d2 = 0.65 )
```

- **Passo 3:** Com o rotor definindo, agora é preciso definir a simulação e seus parâmetros, por padrão o único parâmetro obrigatório é o rotor definido anteriormente na classe “Rotor” contudo, pode-se definir parâmetros para correção de ponta de pá, torção ideal e a discretização da pá.

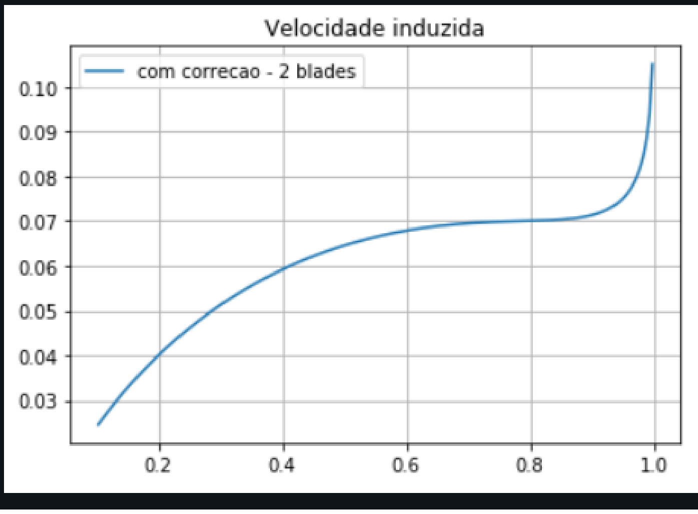
```python
# definindo parametros da simulação

simulacao = bemt.Bemt(rotor = rotorTeste1, correcao = True, twistIdeal = False,
                      elementosPa = 200)
```

- **Passo 4:** Setado os parâmetros da simulação, agora é necessário executar a função solver responsável por gerar o conjunto de *outputs*.

```python
# realizando simulação (outputs)
dct_dr, dcq_dr, vel_ind, cl, k, cp, Fmerit = simulacao.solver()
```

- **Passo 5:** Por fim, com o conjunto de *outputs* é possível plotar os resultados obtidos.

```
# plotando dados
plt.title('Velocidade induzida')
plt.plot(simulacao.r_adim, vel_ind[0,:], label = 'com correcao - 2 blades')
plt.legend()
plt.grid()
plt.show()
```



Para inserir outras curva basta repetir o processo com os novos parâmetros. Para mais detalhes sobre o uso do pacote “aerobemt” e como plotar os resultado mais complexos ver (CORP., 2021). Também será anexado a este tutorial um vídeo explicativo sobre o uso do pacote.

Referências

CORP., T. A. *aero bemt - Tutoriais*. 2021. <https://github.com/marcy3ait/aero-bemt/blob/master/tutoriais.ipynb>.

LEISHMAN, J. G. *Principles of helicopter aerodynamics*. 2nd ed. ed. [S.l.]: Cambridge University Press, 2006.

SEDDON, J. M. *Basic Helicopter Aerodynamics*. 2nd. ed. [S.l.]: Wiley, 2011.

Apêndice

A seguir são apresentadas as equações implementadas no método para a obtenção dos parâmetros de desempenho do rotor de acordo com a metodologia apresentada na literatura (LEISHMAN, 2006).

Inicialmente, deve-se calcular a velocidade induzida, procedimento realizado através de um processo iterativo quando há distribuição linear de torção. O processo inicia-se com o cálculo dessa distribuição através de:

- Distribuição linear de torção

$$\theta = \theta_0 + r_n\theta_{tw} \tag{1}$$

Calculando, a seguir, a torção inicial.

$$\theta^{k=0} = \frac{6C_{TREQ}}{\sigma C_l} - \frac{3}{4}\theta_{tw} + \frac{3}{4}\sqrt{2C_{Treq}} \tag{2}$$

Em seguida, calcula-se a velocidade induzida considerando a presença de correção de ponta de pá ou não. Pelo método de Prandtl:

- Velocidade induzida sem correção (F=1):

$$\lambda(r_n) = \frac{\sigma C_{l\alpha}}{16F}\left(\sqrt{(1 + \frac{32F\theta(r_n)r_n}{\sigma C_{l\alpha}})} - 1\right) \tag{3}$$

- Para a velocidade induzida com correção deve-se avaliar a convergência numérica do valor de F na equação (3), através de:

$$F = \frac{2}{\pi}\arccos(\exp(-f)) \tag{4}$$

$$f = \frac{N_b}{2}\frac{(1 - r_n)}{\lambda(r_n)} \tag{5}$$

Para os casos em que considera-se o afilamento diferente de 1, deve-se incluir a equação da solidez local para o cálculo dos parâmetros que variam em função do raio. Essa grandeza é calculada considerando a ponderação pela tração, a partir de:

$$\sigma_{local} = \frac{(1-\gamma)r_n + r_n}{(1-\gamma)0.75 + \gamma}\sigma_{equi} \tag{6}$$

A partir daí, faz-se o processo iterativo para encontrar o valor convergido da torção inicial, de forma que C_T tende ao valor de $C_{T_{REQ}}$:

$$\theta^{k+1} = \theta_0 + \Delta\theta \tag{7}$$

$$\Delta\theta = \frac{6(C_{TREQ} - C_t)}{\sigma C_{l\alpha}} + \frac{3\sqrt(2)}{4}\frac{(\sqrt(C_{TREQ}) - \sqrt(C_T)}{1} \tag{8}$$

Dessa forma, o gradiente dos parâmetros de desempenho em função do raio adimensionalizado são obtidos a partir das equações abaixo:

- Gradiente do coeficiente de tração

$$\frac{dC_T}{dr} = \frac{\sigma C_{la}}{2}(\theta(r_n)(r_n)^2 - \lambda(r_n)r_n) \tag{9}$$

- Gradiente do coeficiente de torque (numericamente igual ao gradiente de potência total)

$$\frac{dC_Q}{dr} = \frac{dC_{Pi}}{dr} + \frac{dC_{P0}}{dr} \tag{10}$$

- Gradiente de potência induzida

$$\frac{dC_{Pi}}{dr} = \frac{\sigma C_{la}}{2}(\theta(r_n)(r_n)^2 - \lambda(r_n)r_n)\lambda(r_n) \tag{11}$$

- Gradiente de potência de perfil.

$$\frac{dC_{P0}}{dr} = \frac{\sigma}{2}(C_d r_n^3) \tag{12}$$

Para encontrar os coeficientes de tração (C_T), torque (C_Q), potência induzida (C_{Pi}) e potência de perfil (C_{P0}) basta realizar a soma parcial dos diferenciais em cada elemento.

Além disso, os demais coeficientes e parâmetros relevantes ao modelo são calculados por:

- Coeficiente de arrasto de perfil (parabólico)

$$C_d(r_n) = C_{d0} + d1.\left(\theta(r_n) - \frac{\lambda(r_n)}{r_n}\right)^2 + d2.\left(\theta(r_n) - \frac{\lambda(r_n)}{r_n}\right)^3 \tag{13}$$

- Coeficiente de sustentação

$$C_l(r_n) = C_{l\alpha}\left(\theta(r_n) - \frac{\lambda(r_n)}{r_n}\right) \tag{14}$$

- Fator de potência induzida

$$k = \frac{C_{Pi}}{\frac{C_{TREQ}^{3/2}}{2}} \tag{15}$$

- Figura de mérito

$$FM = \frac{\frac{C_{TREQ}^{3/2}}{2}}{k\frac{C_{TREQ}^{3/2}}{2} + C_{P0}} \tag{16}$$

Por fim, para o caso de torção ideal, repete-se o procedimento descrito acima, considerando a distribuição de torção como ideal, de modo que:

$$\theta = \theta_{ideal} = \frac{4C_T}{\sigma C_{l\alpha}} + \sqrt{\frac{C_T}{2}} \tag{17}$$

Sendo assim, a velocidade induzida para este caso é um valor constante, uma vez que, não depende do raio adimensional, sendo calculada por:

$$\lambda(r_n) = \sqrt{\frac{C_T}{2}} \tag{18}$$

A partir daí, é possível obter os demais parâmetros da mesma forma que foi empregada no procedimento para a distribuição linear de torção.