Guia de Uso do Programa *eulerblock*

Maj. Eng. **Ney** Rafael Sêcco, ney@ita.br
Instituto Tecnológico de Aeronáutica
Divisão de Engenharia Aeronáutica
Departamento de Projetos

August 27, 2025

1 Introdução

Esse código analisa aerofólios segundo a equação de Euler para otimizar aerofólios para o regime transônico.

2 Descrição das códigos fornecidos

2.1 Módulos

Os seguintes módulos são fornecidos:

1. `eulerblock.exe`: código em Fortran que resolve as equações de Euler em uma malha-O estruturada bidimensional ao redor de um aerofólio. Esse programa utiliza uma metodologia de volumes finitos de segunda ordem, com fluxos calculados por meio da média de Roe. A convergência é feita por meio de Runge-Kutta de ordem 5 com passo de tempo local baseado em CFL. Esse código espera que a malha em formato PLOT3D esteja presente no arquivo `airfoil.xyz`. As condições da simulação devem estar escritas no arquivo `settings.txt`. Após a simulação, o código gravará os resultados nos arquivos `wall.dat` (informações das fronteiras do domínio) e `solution.vtk` (informações de campo em formato compatível com o software Paraview ou VisIt). Esse código aplica o método adjunto para calcular as derivadas dos coeficientes aerodinâmicos com respeito aos nós da malha e condições de contorno. Tais derivadas são guardadas no arquivo `derivatives.dat` após a execução do código;

2. `gridgen2d.py`: Gerador de malhas 2D estruturadas por meio de equações hiperbólicas. Esse código recebe a geometria do perfil para gerar a malha no espaço;

3. `gridwarp.py`: Deformador de malha 2D pelo método de ponderação pelo inverso da distância. Esse código atualiza a malha do espaço em função de deformações nos nós da superfície do aerofólio;

4. `euler_mod.py`: Módulo em Python que realiza a interface com o código em Fortran. Esse módulo chama o gerador de malha para escrever o arquivo de malha (`airfoil.xyz`) e também escreve o arquivo com as condições de simulação do código Euler (`settings.txt`);

5. `airfoil_mod.zip`: Módulo de parametrização de aerofólios por meio da formulação CST e NACA. Esse módulo Python precisa ser instalado separamente por meio do *pip*.

2.2 Pacotes externos

O módulo em Python `airfoil_mod`, utilizado para parametrização de aerofólios, precisa ser instalado separadamente por meio do *pip*. Isso pode ser feito por meio das seguintes etapas:

1. Descompacte o arquivo `airfoil_mod.zip` em algum diretório de seu computador (ou clone o repositório a partir desse link: airfoil_mod).

2. Abra o explorador de arquivos e navegue até a pasta descompactada;

3. Copie o endereço da pasta (clique na barra de endereço e aperte CTRL+C);

4. Abra o Menu Iniciar e procure por "anaconda prompt" ou "anaconda powershell";

5. No terminal, use o comando `cd` para navegar até a pasta dos códigos. Basta digitar "cd" e depois clicar no ícone do topo esquerdo da janela e clicar em "Editar" > "Colar". Aperte <ENTER> em seguida.

6. Agora execute o comando de instalação:

   ```
   pip install .
   ```

2.3 Scripts

As principais tarefas desse laboratório poderão ser resolvidas por meio da execução dos seguintes scripts contidos na pasta `run_cases`:

- `01_airfoil_plotter`: Esse script permite analisar aerofólios de forma interativa. Ao executar o código, o usuário observará uma janela que facilita a interface com o código Euler.

  Na lateral esquerda da janela estão campos para selecionar o Mach da simulação, o número máximo de iterações, o CFL (número de Courant) para controle do passo no tempo e a tolerância dos resíduos desejada.

No canto inferior da janela, o usuário pode escolher "no" para que o processo iterativo seja inicializada assumindo escoamento uniforme, ou escolher "yes" para utilizar as informações de campo já guardadas no arquivo `solution.vtk` para reinicializar a solução a partir de um caso anterior (isso pode acelerar a convergência).

No centro da janela há barras interativas para ajustar os parâmetros CST do intradorso (`Au`) e do extradorso (`Al`) do aerofólio, bem como o ângulo de ataque da simulação.

Na parte inferior da janela há três botões: 1) *Export* para criar o arquivo `airfoil.dat` com as coordenadas do perfil no formato esperado pelo Xfoil, 2) *Run* para analisar o aerofólio com o código Euler e 3) *Reset* para colocar as barras ajustáveis de volta aos valores padrão.

- `02_single_run`: Script que mostra como executar o código Euler sem a interface gráfica. Esse script automaticamente gera o arquivo `airfoil.dat` com as coordenadas do perfil no formato esperado pelo Xfoil.

- `03_optimize`: Script que realiza a otimização de aerofólio para uma dada condição de escoamento fornecida pelo usuário. A definição do problema de otimização é:

$$\begin{aligned} \min \quad & c_d \\ \text{w.r.t.} \quad & \text{CST coefficients}(A_l, A_u),\ \alpha \\ \text{s.t.} \quad & c_\ell = c_{\ell\text{ref}} \\ & (t/c) \geq (t/c)_{\text{ref}} \\ & A_{l,\text{lower}} \preceq A_l \preceq A_{l,\text{upper}} \\ & A_{u,\text{lower}} \preceq A_u \preceq A_{u,\text{upper}} \end{aligned} \tag{1}$$

  Os valores de $c_{\ell\text{ref}}$ e $(t/c)_{\text{ref}}$ são definidos pelo usuário no início script. O otimizador utilizado é o SLSQP, com gradientes estimados por meio do método adjunto. **Esse script pode demorar horas para executar.**

Sugere-se que os scripts sejam executados por meio do prompt de comando para permitir o monitoramento das iterações do código Euler. Siga os seguintes passos:

1. Abra o explorador de arquivos e navegue até a pasta que contém os códigos fornecidos;

2. Copie o endereço da pasta (clique na barra de endereço e aperte CTRL+C);

3. Abra o Menu Iniciar e procure por "anaconda prompt" ou "anaconda powershell";

4. No terminal, use o comando `cd` para navegar até a pasta dos códigos. Basta digitar "cd" e depois clicar no ícone do topo esquerdo da janela e clicar em "Editar" > "Colar". Aperte <ENTER> em seguida.

5. Agora execute o script chamando o Python. Por exemplo, para executar o script `single_run.py` digite:

   ```
   python single_run.py
   ```

O script `optimize.py` pode demorar horas até convergir. Enquanto esse script roda em um prompt de comando, é possível acompanhar o progresso da otimização abrindo um novo prompt no mesmo diretório e executando o script `plot_history.py`. Esse script também gera o arquivo `airfoil.dat` que pode ser analisado no Xfoil.

3 Paraview

Quem estiver com o Paraview (`https://www.paraview.org/download/`) instalado no computador pode observar a solução no domínio carregando o arquivo `paraview_state_windows.py` com a opção "Load State". Esse arquivo sempre carregará o arquivo `solution.vtk` da pasta `02_single_run`.

4 Outras dicas

- Caso a otimização seja interrompida por algum motivo, você pode reinicializá-la aproveitando o histórico anterior com o seguinte procedimento:

  1. Faça uma cópia do arquivo `opt_results.pickle` presente na pasta do código e renomeie essa cópia para `opt_results_copy.pickle`;
  2. Abra o arquivo `optimize.py` e, na linha 91, substitua:

     `reload_opt = None`

     por:

     `reload_opt = 'opt_results_copy.pickle'`;
  3. Execute novamente o script `optimize.py`.

- Tentem alterar o parâmetro `ftol` do arquivo `optimize.py` para uma tolerância menor (por exemplo $1.0 \cdot 10^{-5}$) para obter aerofólios mais otimizados. No entanto, isso pode aumentar o tempo de otimização. Vocês também podem alterar os batentes da otimização alterando as variáveis `Al_lower`, `Al_upper`, `Au_lower`, `Au_upper`.