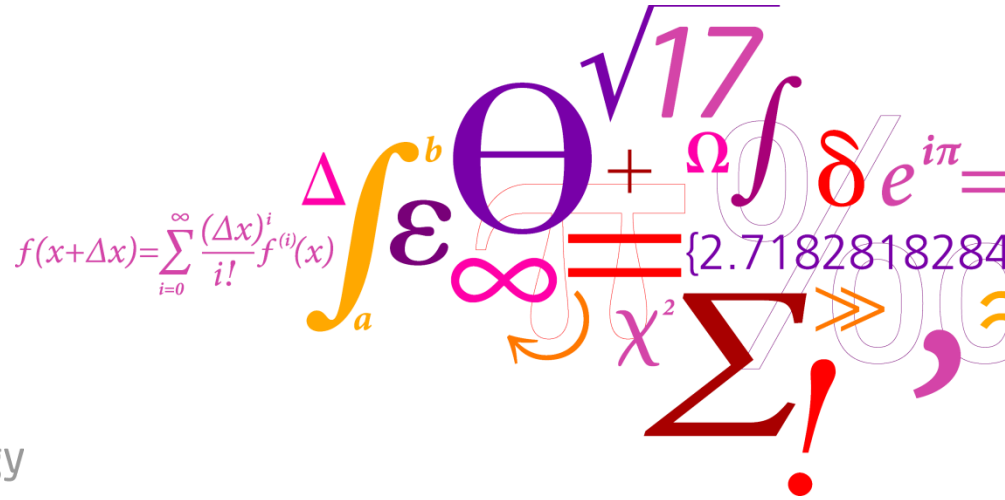
Coupling the Navier Stokes actuator line model with the aeroelastic solver HAWC2 – work in progress

Niels Troldborg and Joachim Heinz

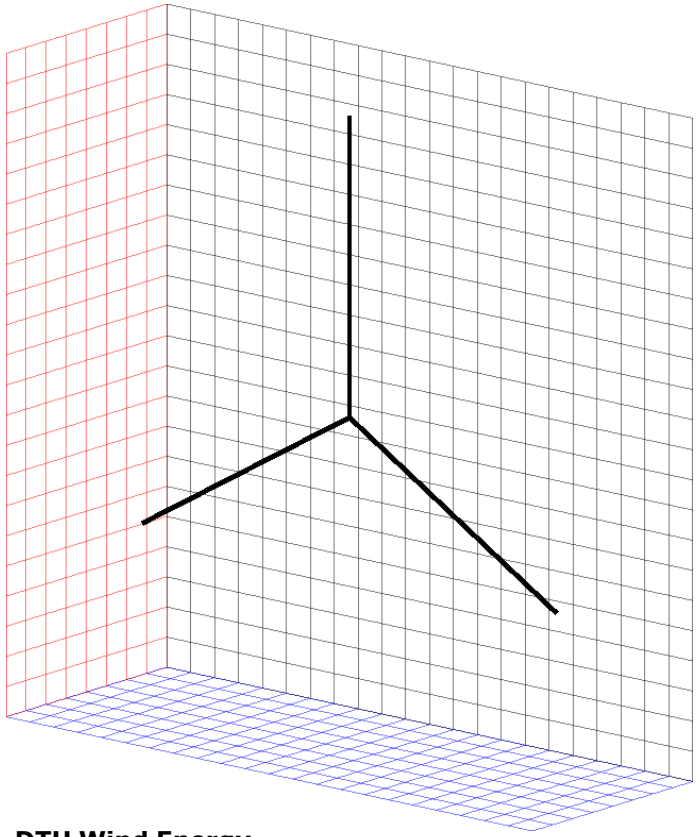
Wind Energy Department, DTU Wind Energy, DK-4000 Roskilde, Denmark



Risø DTU
National Laboratory for Sustainable Energy

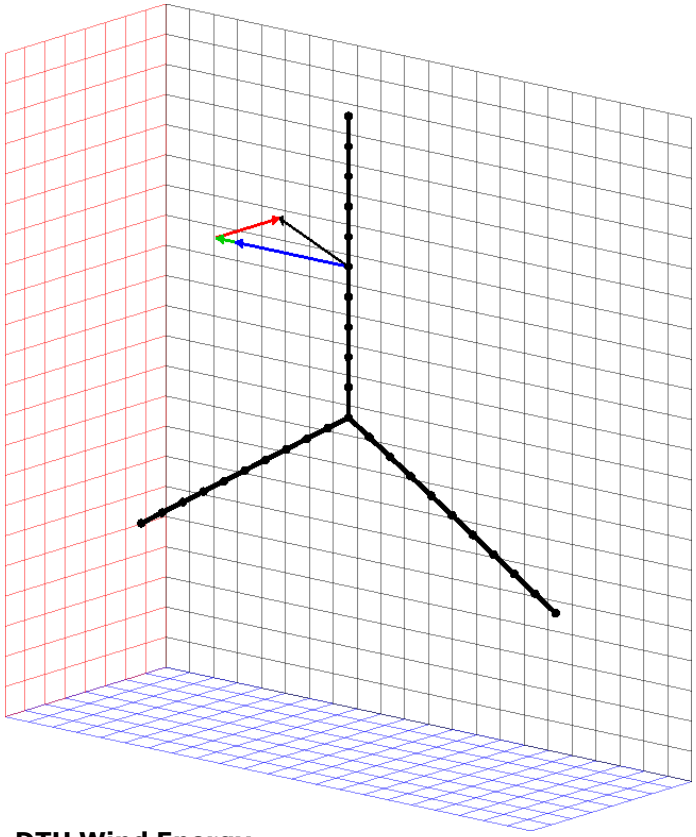
Actuator Line/Navier-Stokes simulations

- Blades represented as lines
- Flow field determined from 3D N-S simulations



Actuator Line/Navier-Stokes simulations

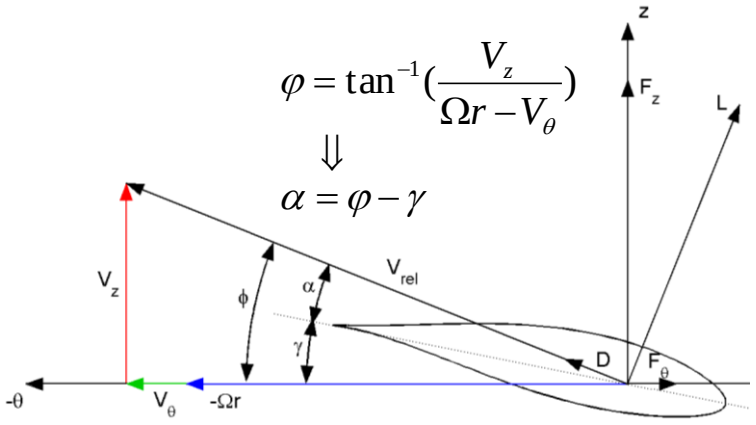
- Blades represented as lines
- Flow field determined from 3D N-S simulations
- Aerodynamic forces at each blade section determined from 2D airfoil data



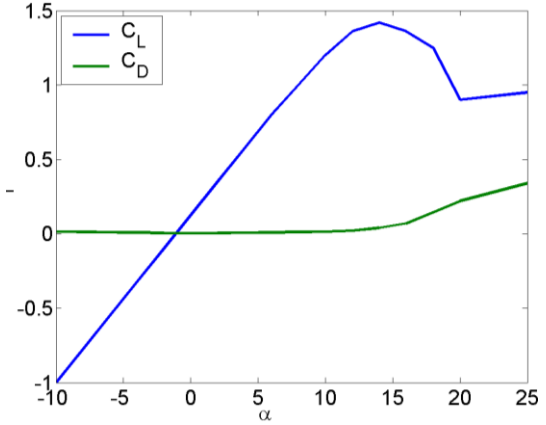
$$\varphi = \tan^{-1}\left(\frac{V_z}{\Omega r - V_\theta}\right)$$

$$\Downarrow$$

$$\alpha = \varphi - \gamma$$

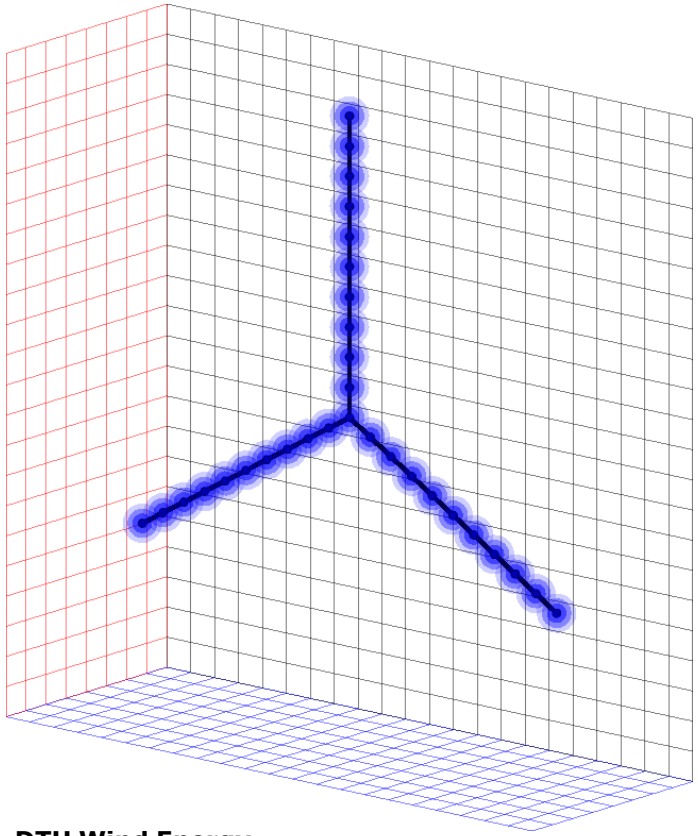


$$\mathbf{f} = \begin{pmatrix} L \\ D \end{pmatrix} = \tfrac{1}{2}\rho V_{rel}^2 c \begin{pmatrix} C_L(\alpha)\mathbf{e}_L \\ C_D(\alpha)\mathbf{e}_D \end{pmatrix}$$



Actuator Line/Navier-Stokes simulations

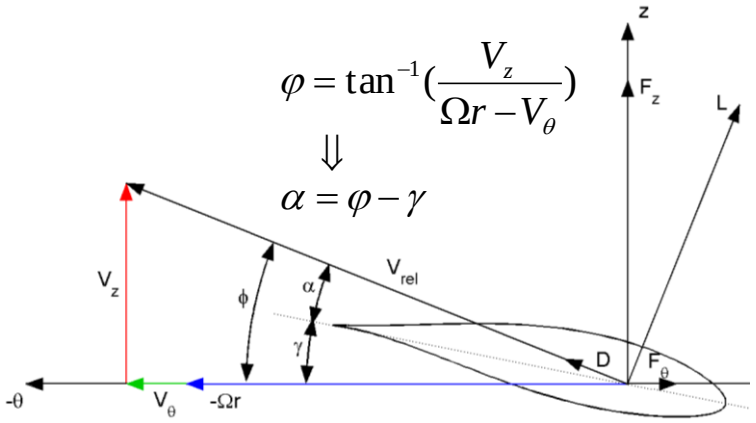
- Blades represented as lines
- Flow field determined from 3D N-S simulations
- Aerodynamic forces at each blade section determined from 2D airfoil data
- Blade forces transferred to N-S solver using Gaussian smearing to avoid singular behaviour



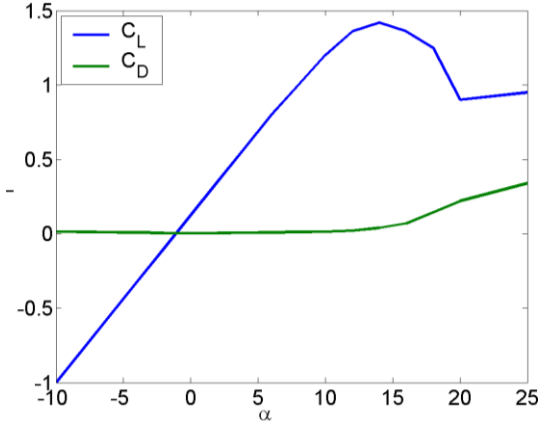
$$\varphi = \tan^{-1}\left(\frac{V_z}{\Omega r - V_\theta}\right)$$

$$\Downarrow$$

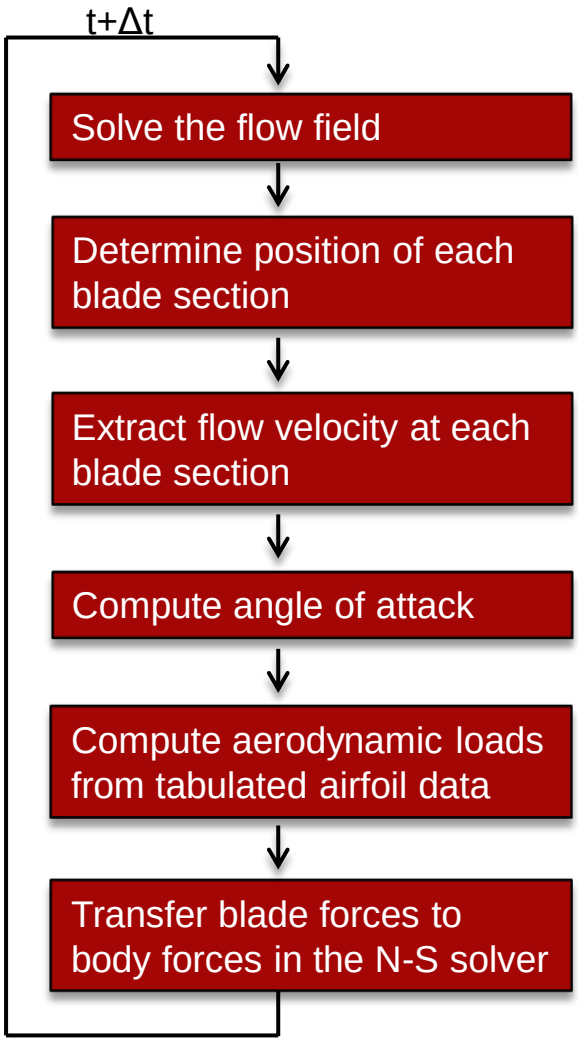
$$\alpha = \varphi - \gamma$$



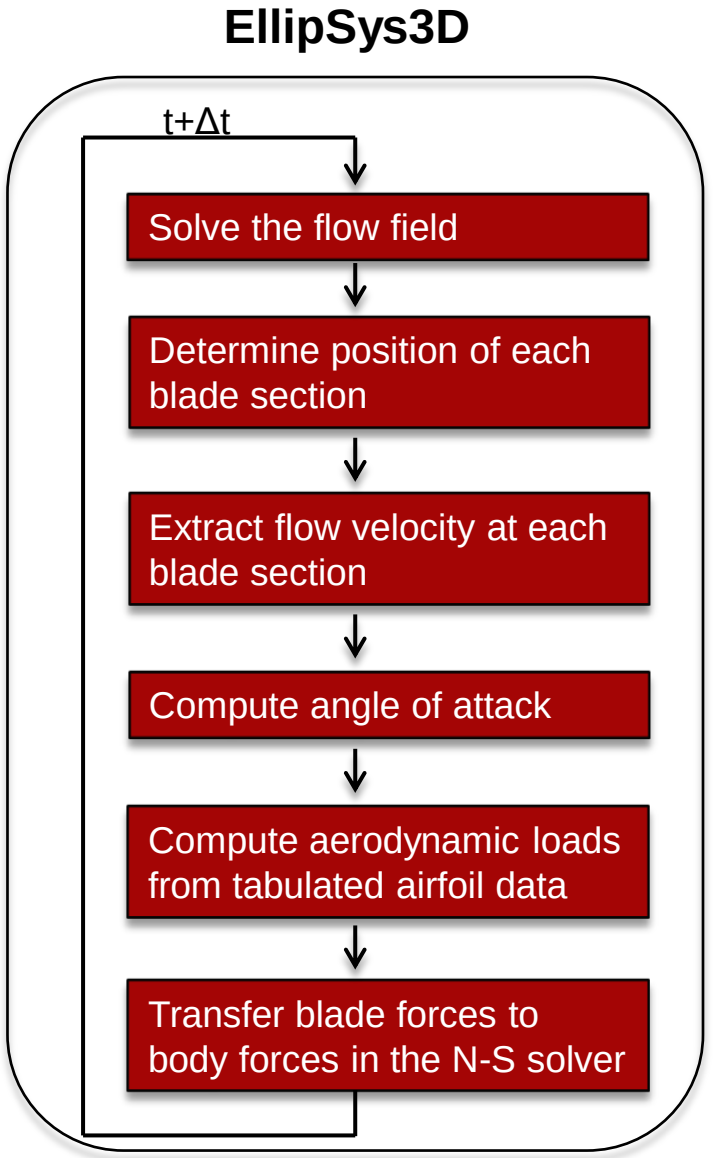
$$\mathbf{f} = \begin{pmatrix} L \\ D \end{pmatrix} = \tfrac{1}{2}\rho V_{rel}^2 c \begin{pmatrix} C_L(\alpha)\mathbf{e}_L \\ C_D(\alpha)\mathbf{e}_D \end{pmatrix}$$



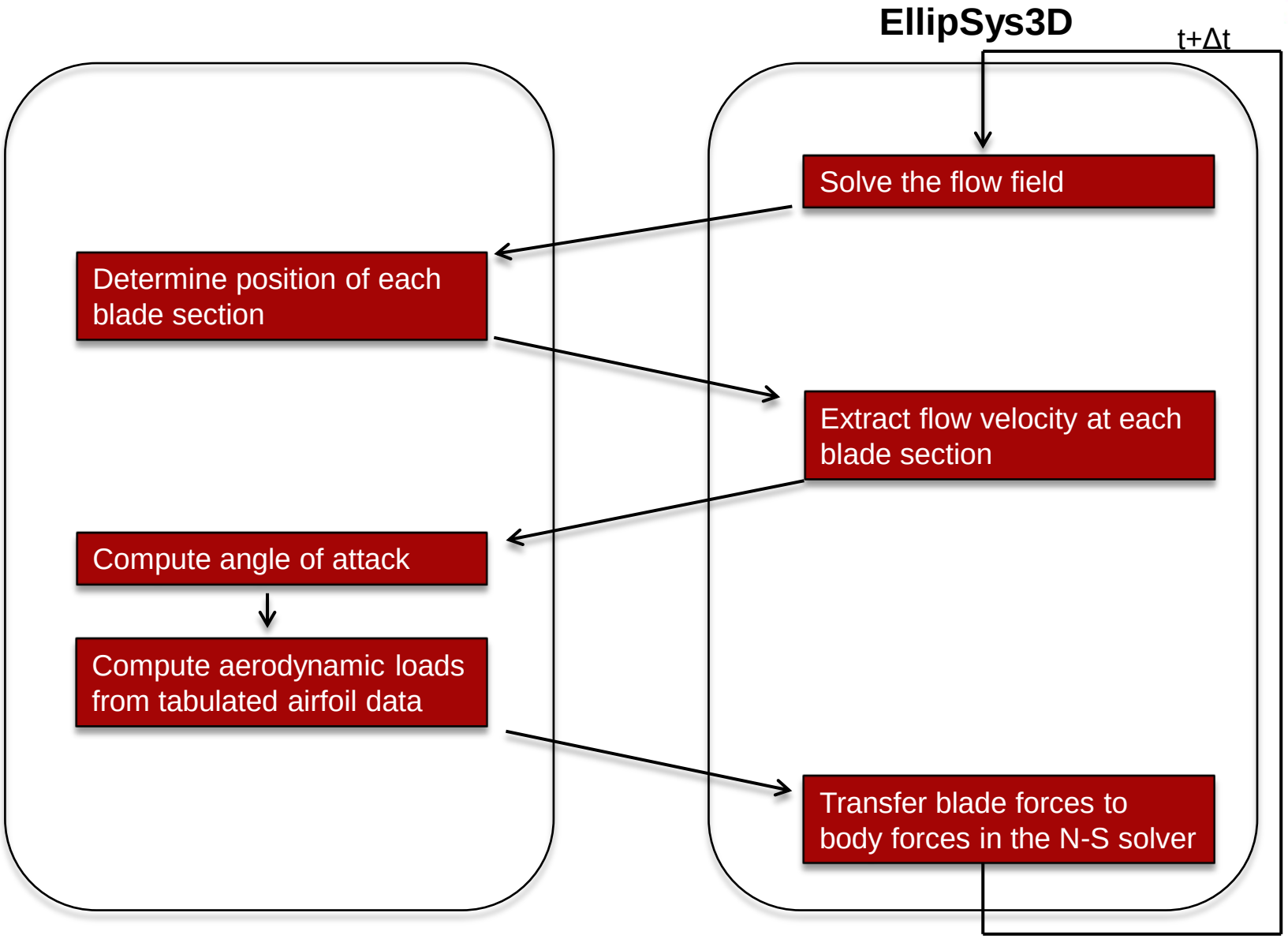
Actuator Line/Navier-Stokes simulations



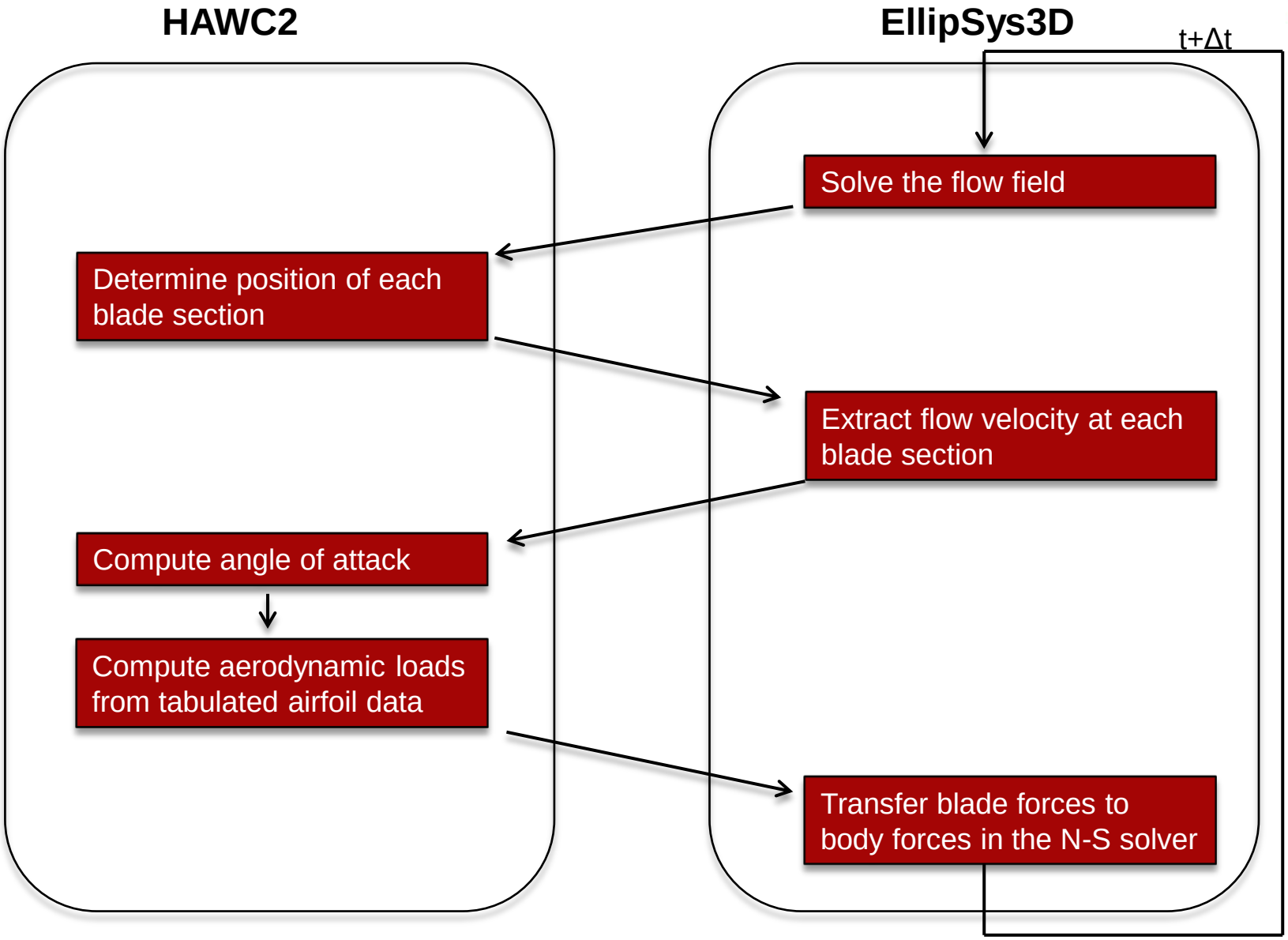
Actuator Line/Navier-Stokes simulations



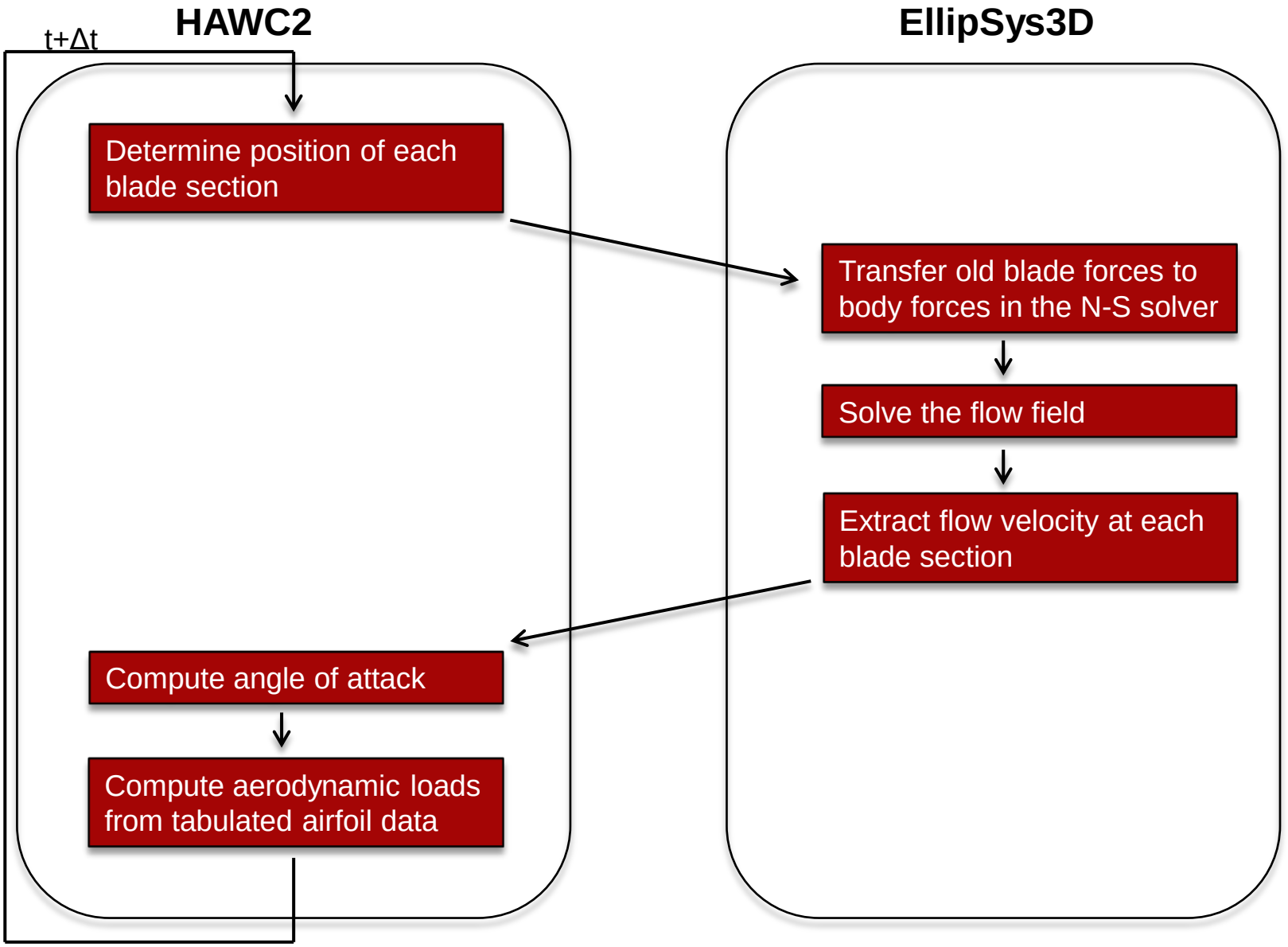
Actuator Line/Navier-Stokes simulations



HAWC2/Actuator Line/Navier-Stokes simulations



HAWC2/Actuator Line/Navier-Stokes simulations



Generic coupling framework

Step 1:

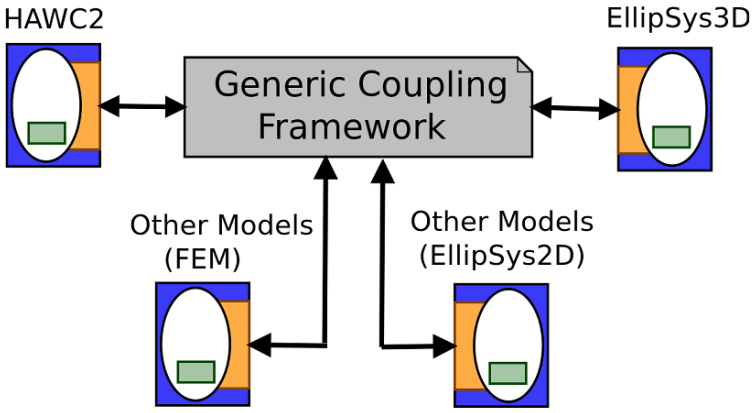
- Wrap the programs participating in the coupling

Step 2:

- Import the wrapped programs into python

Step 3:

- Python orchestras the execution of the programs and organizes the input/output handling via the common script

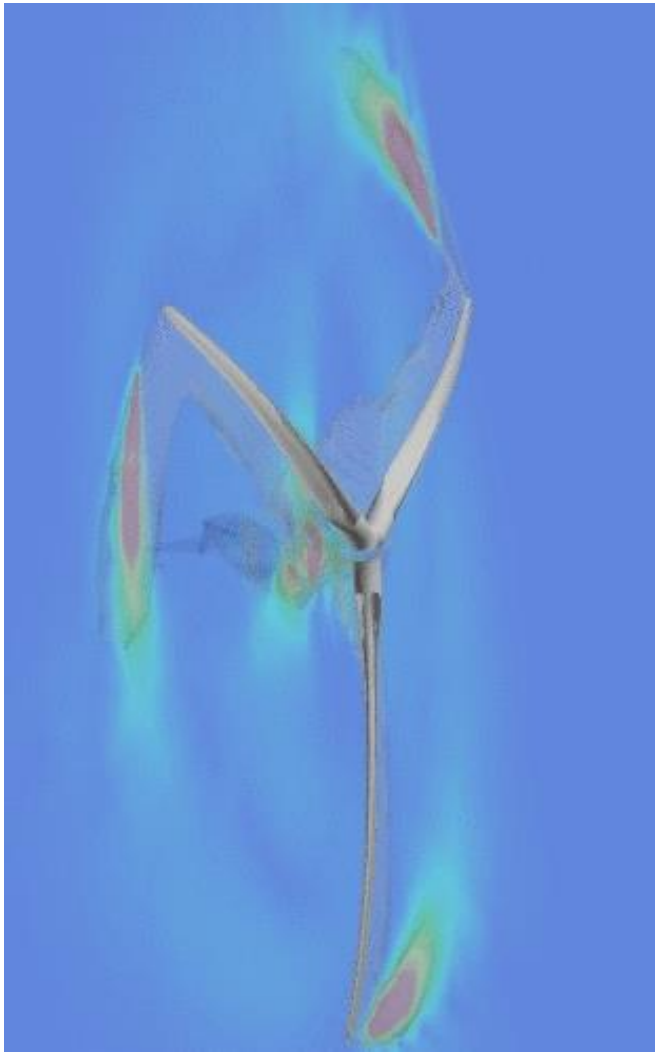


Generic coupling framework

- OpenMDAO (Multidisciplinary Design Analysis and Optimization)
  - A framework for executing and connecting different types of codes and optimizers
  - Open source (written in Python)
  - Minimally intrusive
    - solvers are kept as independent entities
    - leave participating codes unchanged
  - Generic
    - standardized interface function
    - models can easily be exchanged or added
  - Flexible
    - connect codes written in different languages
    - connect codes run on different platforms

Generic coupling framework

- OpenMDAO (Multidisciplinary Design Analysis and Optimization)
  - A framework for executing and connecting different types of codes and optimizers
  - Open source (written in Python)
  - Minimally intrusive
    - solvers are kept as independent entities
    - leave participating codes unchanged
  - Generic
    - standardized interface function
    - models can easily be exchanged or added
  - Flexible
    - connect codes written in different languages
    - connect codes run on different platforms

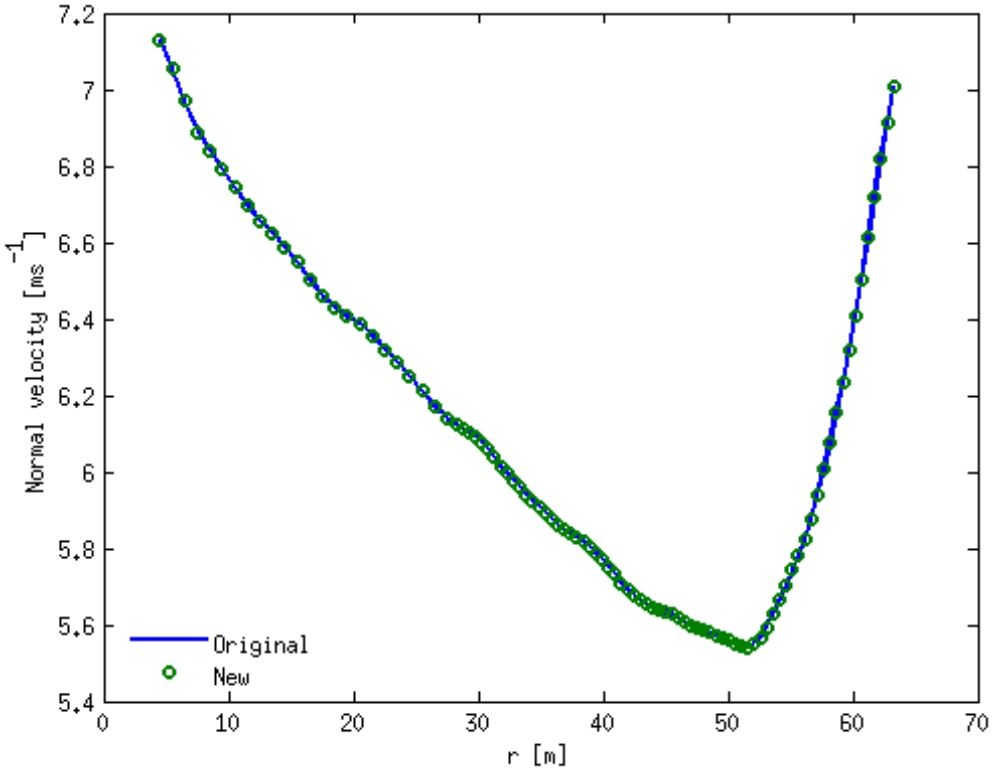


Simulating the NREL 5MW wind turbine

- Wind speed: V_∞ = 8 m/s
- Rotational speed Ω = 0.964 rad/s
- Forces prescribed according to results from a full rotor simulation
- Blade section coordinates and blade loading provided by external routine (HAWC2 emulator)

Simulating the NREL 5MW wind turbine

- Wind speed: V_∞ = 8 m/s
- Rotational speed Ω = 0.964 rad/s
- Forces prescribed according to results from a full rotor simulation
- Blade section coordinates and blade loading provided by external routine (HAWC2 emulator)



Conclusions

- The actuator line method has been included in the OpenMDAO framework
- The actuator line method has been modified to make it as general as HAWC2, i.e. can handle a multiple of different types of turbines.
- First test case shows good results