

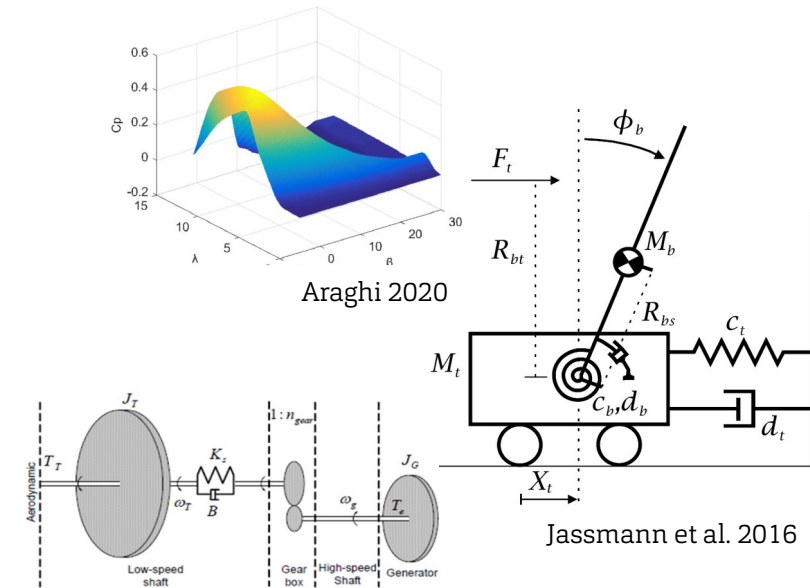
**Symbolic  
Flexible  
Multibody**  
Models for  
Wind Turbine  
Controller  
Design  
and Analysis

Prof. Dr.-Ing. Jens Geisler  
May 16, 2021



# Motivation

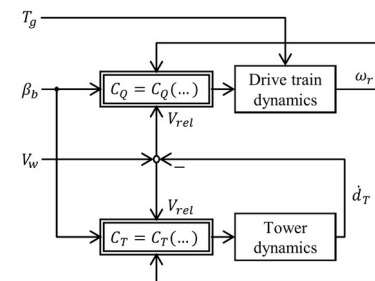
- Wind turbine development relies on models
  - Esp. simplified dynamic models
  - Various levels of fidelity, depending on task
- Currently, usually one-off developments for a specific purpose
  - Based on a heuristic structure and simplifications
  - Or linearizations from high detail simulators
- Change management and traceability difficult
- **One source for whole family of models desirable**



Bassi and Mobarak, 2017

$$\begin{bmatrix} \dot{\omega}_t \\ \dot{\omega}_g \\ T_{tw} \\ T_g \\ \beta \end{bmatrix} = \begin{bmatrix} 0 & 0 & -\frac{N_g}{J} & 0 & 0 \\ 0 & 0 & \frac{1}{J_g} & -\frac{1}{J_g} & 0 \\ K_s N_g & -K_s & -\left(\frac{N_g^2 B_s}{J} + \frac{B_s}{J_g}\right) & \frac{B_s}{J_g} & 0 \\ 0 & 0 & 0 & -\frac{1}{J_g} & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{\tau} \end{bmatrix} \begin{bmatrix} \omega_t \\ \omega_g \\ T_{tw} \\ T_g \\ \beta \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \frac{1}{J_g} & 0 \\ 0 & 0 \end{bmatrix} T_{s,ref} + \begin{bmatrix} \frac{1}{J_t} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} T_t$$

Ghanbarpour et al 2020

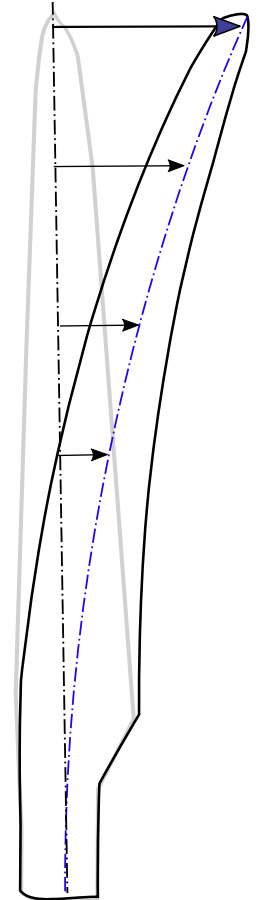


# Modular Modeling with Kane's Method

- Formalism for deriving **generalized equations of motion**
  - No constraint equations
  - Only as many ODEs as DOF
- **Intuitive, modular definition of the System**
  - User defines pose and inertial parameters per body
  - Automatic formulation of Newton-Euler-equations
  - Automatic removal of constraints
  - Enabled by powerful computer algebra systems (CAS)
- **Symbolic equations of motion**
  - Variable (symbolic) parameters
  - Can be further processed in many different ways

# Elastic / Flexible Bodies

- Floating frame of reference
- Superposition of rigid (undeformed) and flexible body motion
- Flexible coordinates may be finite elements or modes
- Standard Input Data (SID due to Schwertassek and Wallrapp)
  - Linear-quadratic representation
  - Includes centrifugal stiffening and gravitational softening
  - Standardized data format for interfacing FEM tools with dynamic modeling tools
  - Offloading of computations to pre-simulation stage
  - Supported e.g. by ANSYS and Simpack



- **Comparison of three model variations with OpenFAST**
  - 6 DOF: tower fore-aft & side-side, generator & hub speed, collective blade flap- & edge-wise
  - 4 DOF: tower fore-aft, generator & hub speed, collective blade flap-wise
  - 2 DOF: tower fore-aft, hub speed
  - All blade motions are collective

# Results – 6DOF Model

```
/* 1 Tower */
load(filename_merge(load_pathname, "tw_sid.mac"));
elastic_dof[1]: [q[1], q[2]];
ebody[1]: tower;
T0G[1]: Tdisp(0, 0, 0);

/* 2 Nacelle */
mass[2]: NacMass;
Ixx[2]: NacXIner;
Iyy[2]: NacYIner;
Izz[2]: NacZIner;
BodyRef[2]: 1;
/* TrefG[2]: Tdisp(q[1], q[2], 0).Tdisp(NacCMxn, NacCMyn, NacCMzn); */
TrefG[2]: Telast(1, 1).Tdisp(NacCMxn, NacCMyn, NacCMzn);
/* TrefG[2]: Tdisp(q[1], q[2], 0).Tdisp(NacCMxn, NacCMyn, NacCMzn).Troty_lin(q[1]*TwTrans2Roll).Trotx_lin(-q[2]*TwTrans2Roll); */

/* 3 Hub */
mass[3]: HubMass;
Ixx[3]: HubIner;
Iyy[3]: HubIner;
Izz[3]: HubIner;
BodyRef[3]: 2;
TrefG[3]: Tdisp(HubCM+OverHang-NacCMxn, -NacCMyn, Twr2Shft-NacCMzn).Trotx(q[3]);

/* 4,5,6 Blades */
load(filename_merge(load_pathname, "bd_sid.mac"));

/* Blade 1 */
elastic_dof[4]: [q[4], q[5]];
ebody[4]: blade;
BodyRef[4]: 3;
TrefG[4]: Tdisp(-HubCM, 0, 0).Trotz(theta);

/* Blade 2 */
elastic_dof[5]: [q[4], q[5]];
```

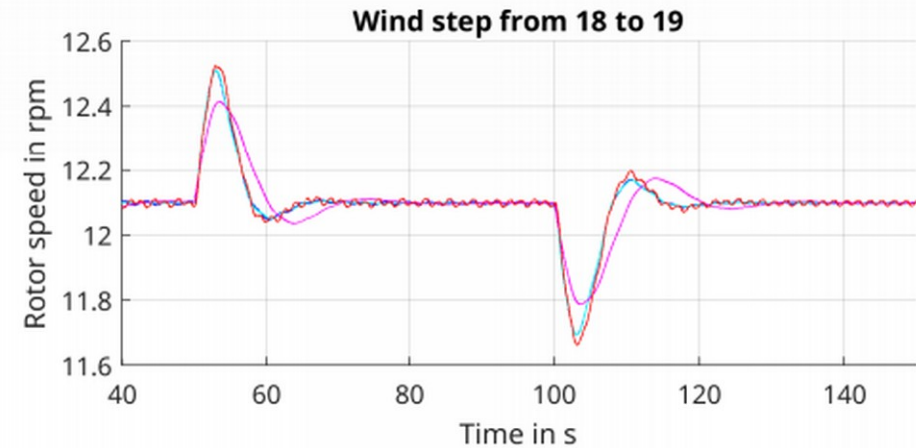
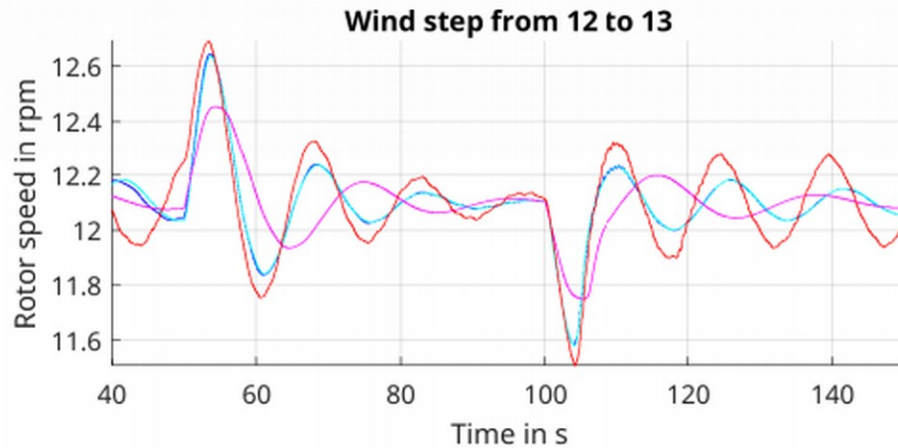
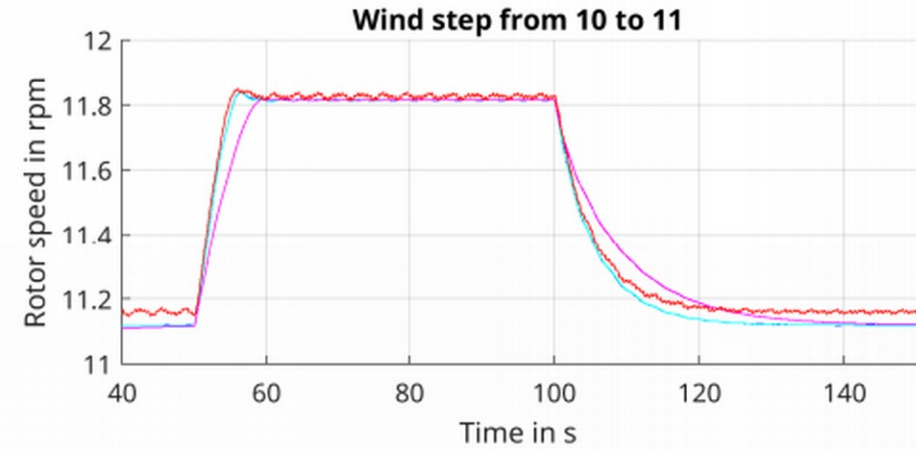
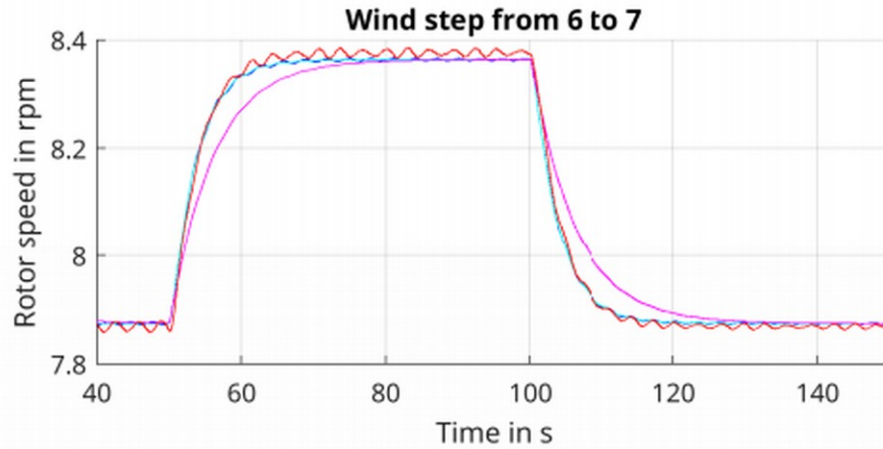
# Results – Generalized Equations of Motion

$$\begin{aligned}
 & \ddot{q}_1 m_{e,tow,0}[1,1] + q_1 K_{e,tow,0}[1,1] + \dot{q}_1 D_{e,tow,0}[1,1] + q_1 g C_{t,tow,1}[1,1,3] + (-3 \ddot{q}_5 C_{t,bld,0}[2,2] - 3 \ddot{q}_4 C_{t,bld,0}[1,2]) \sin \vartheta + \dots \\
 & \quad + (3 \ddot{q}_5 C_{t,bld,0}[2,1] + 3 \ddot{q}_4 C_{t,bld,0}[1,1]) \cos \vartheta + 3 \ddot{q}_1 m_{bld} + \ddot{q}_1 m_{nac} + \ddot{q}_1 m_{hub} - F_{hub} \\
 & \quad \ddot{q}_2 m_{e,tow,0}[2,2] + q_2 K_{e,tow,0}[2,2] + \dot{q}_2 D_{e,tow,0}[2,2] + q_2 g C_{t,tow,1}[2,2,3] + 3 \ddot{q}_2 m_{bld} + \ddot{q}_2 m_{nac} + \ddot{q}_2 m_{hub} \\
 & \quad \frac{1}{i_{gear}} ((3 \ddot{q}_3 i_{gear} I_{e,bld,0}[2,2] - 3 \ddot{q}_3 i_{gear} I_{e,bld,0}[1,1]) \sin^2 \vartheta + \dots \\
 & + (-3 \ddot{q}_5 i_{gear} C_{r,e,bld,0}[2,2] - 3 \ddot{q}_4 i_{gear} C_{r,e,bld,0}[1,2]) \sin \vartheta + (3 \ddot{q}_5 i_{gear} C_{r,e,bld,0}[2,1] + 3 \ddot{q}_4 i_{gear} C_{r,e,bld,0}[1,1]) \cos \vartheta + \dots \\
 & \quad + 3 \ddot{q}_3 i_{gear} I_{e,bld,0}[1,1] - i_{gear} Trot + \ddot{q}_3 i_{gear} HubIner + (q_3 K_{hub,gen} + \dot{q}_3 D_{hub,gen}) i_{gear} - q_6 K_{hub,gen} - \dot{q}_6 D_{hub,gen} \\
 & \quad (3 \dot{q}_3^2 q_5 O_{e,bld,1}[2,1,2] + 3 \dot{q}_3^2 q_4 O_{e,bld,1}[1,1,2]) \sin^2 \vartheta + \dots \\
 & + ((-3 \dot{q}_3^2 q_5 O_{e,bld,1}[2,1,4] - 3 \dot{q}_3^2 q_4 O_{e,bld,1}[1,1,4]) \cos \vartheta - 3 \ddot{q}_1 C_{t,bld,0}[1,2] - 3 \ddot{q}_3 C_{r,e,bld,0}[1,2]) \sin \vartheta + \dots \\
 & \quad + (3 \dot{q}_3^2 q_5 O_{e,bld,1}[2,1,1] + 3 \dot{q}_3^2 q_4 O_{e,bld,1}[1,1,1]) \cos^2 \vartheta + \dots \\
 & \quad + (3 \ddot{q}_1 C_{t,bld,0}[1,1] + 3 \ddot{q}_3 C_{r,e,bld,0}[1,1]) \cos \vartheta - 3 F_{e,flap} + \dots \\
 & \quad + 3 \ddot{q}_5 m_{e,bld,0}[1,2] + 3 \ddot{q}_4 m_{e,bld,0}[1,1] + 3 q_5 K_{e,bld,0}[1,2] + 3 q_4 K_{e,bld,0}[1,1] + 3 \dot{q}_4 D_{e,bld,0}[1,1] \\
 & \quad (3 \dot{q}_3^2 q_5 O_{e,bld,1}[2,2,2] + 3 \dot{q}_3^2 q_4 O_{e,bld,1}[1,2,2]) \sin^2 \vartheta + \dots \\
 & + ((-3 \dot{q}_3^2 q_5 O_{e,bld,1}[2,2,4] - 3 \dot{q}_3^2 q_4 O_{e,bld,1}[1,2,4]) \cos \vartheta - 3 \ddot{q}_1 C_{t,bld,0}[2,2] - 3 \ddot{q}_3 C_{r,e,bld,0}[2,2]) \sin \vartheta + \dots \\
 & \quad + (3 \dot{q}_3^2 q_5 O_{e,bld,1}[2,2,1] + 3 \dot{q}_3^2 q_4 O_{e,bld,1}[1,2,1]) \cos^2 \vartheta + \dots \\
 & \quad + (3 \ddot{q}_1 C_{t,bld,0}[2,1] + 3 \ddot{q}_3 C_{r,e,bld,0}[2,1]) \cos \vartheta - 3 F_{e,edge} + 3 \ddot{q}_5 m_{e,bld,0}[2,2] + \dots \\
 & \quad + 3 \ddot{q}_4 m_{e,bld,0}[2,1] + 3 q_5 K_{e,bld,0}[2,2] + 3 q_4 K_{e,bld,0}[2,1] + 3 \dot{q}_5 D_{e,bld,0}[2,2] \\
 & \quad \frac{i_{gear} T_{gen} + \dot{q}_6 i_{gear} I_{gen} + (-q_3 K_{hub,gen} - \dot{q}_3 D_{hub,gen}) i_{gear} + q_6 K_{hub,gen} + \dot{q}_6 D_{hub,gen}}{i_{gear}}
 \end{aligned}$$

- **Comparison of three model variations with OpenFAST**
  - 6 DOF: tower fore-aft & side-side, generator & hub speed, collective blade flap- & edge-wise
  - 4 DOF: tower fore-aft, generator & hub speed, collective blade flap-wise
  - 2 DOF: tower fore-aft, hub speed
  - All blade motions are collective
- **Standalone, high performance simulation program**
  - Special rotor effective aerodynamics model with damping and blade mode excitation
  - Interface to DISCON.dll
  - Read OpenFAST parameters and wind data (rotor effective)
  - Write OpenFAST binary output
- **Simulation of wind steps up and down 1 m/s**

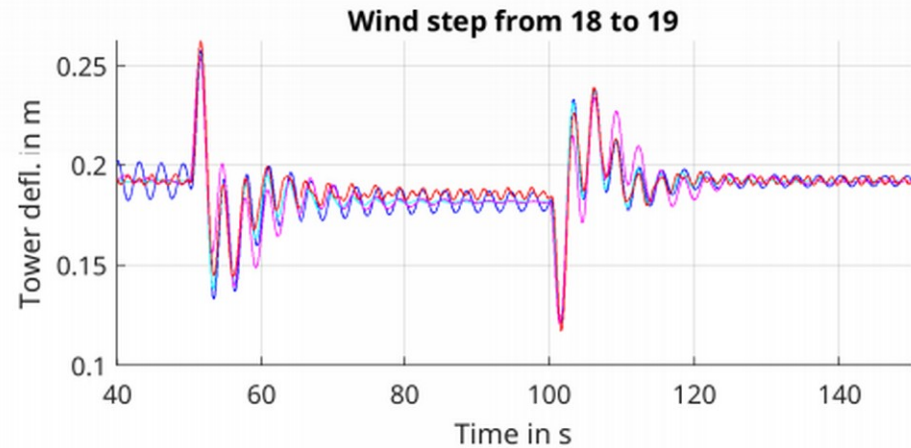
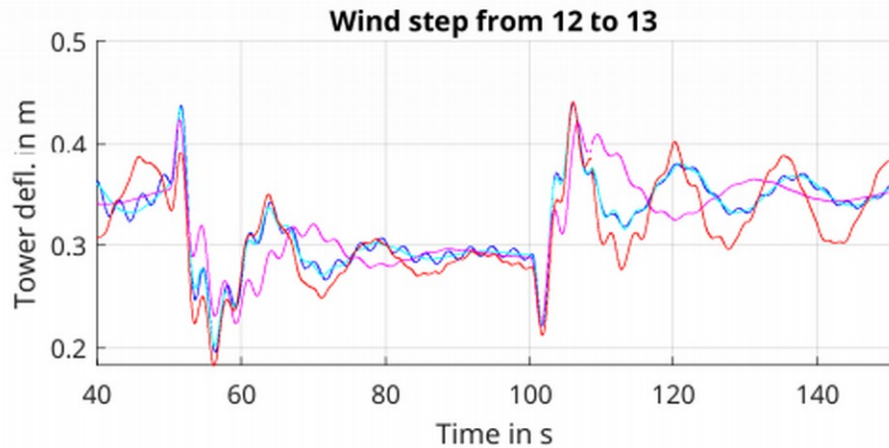
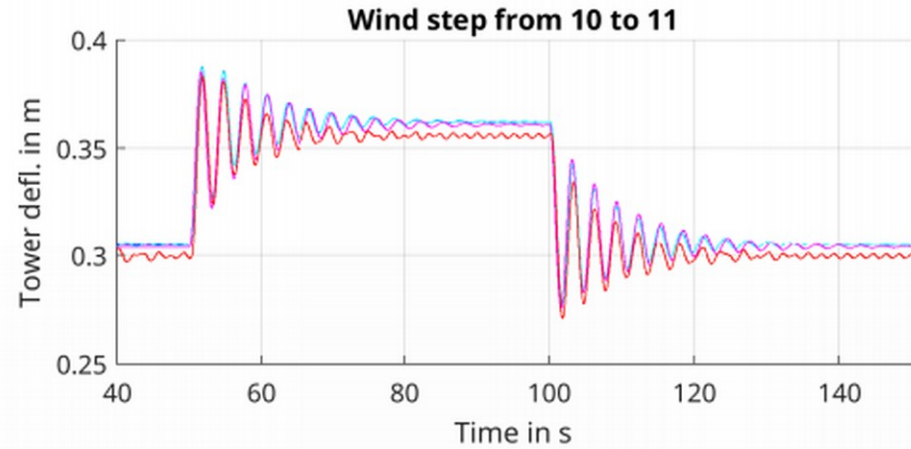
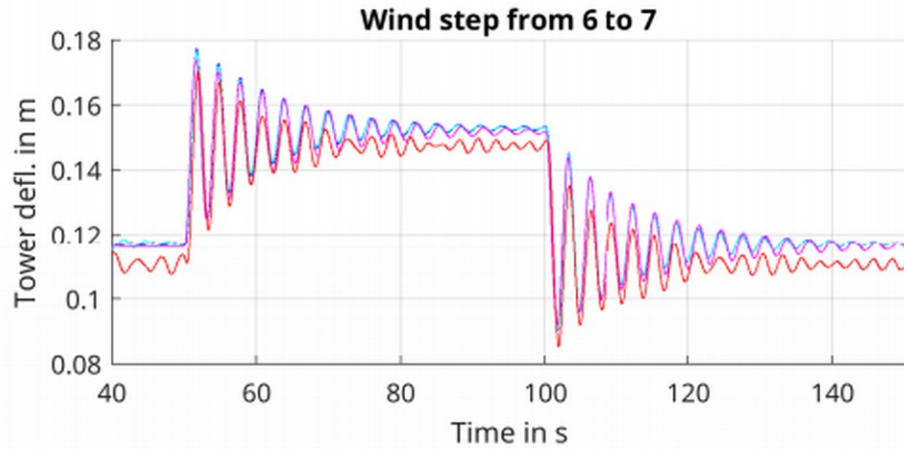


# Results – Rotor Speed



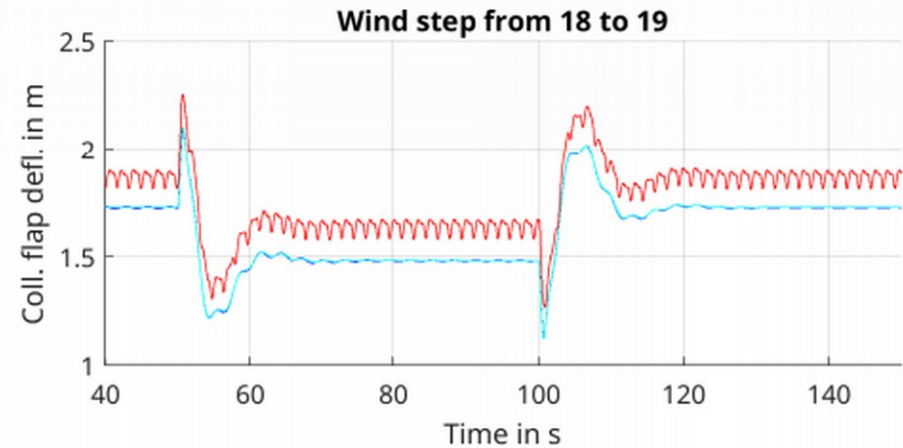
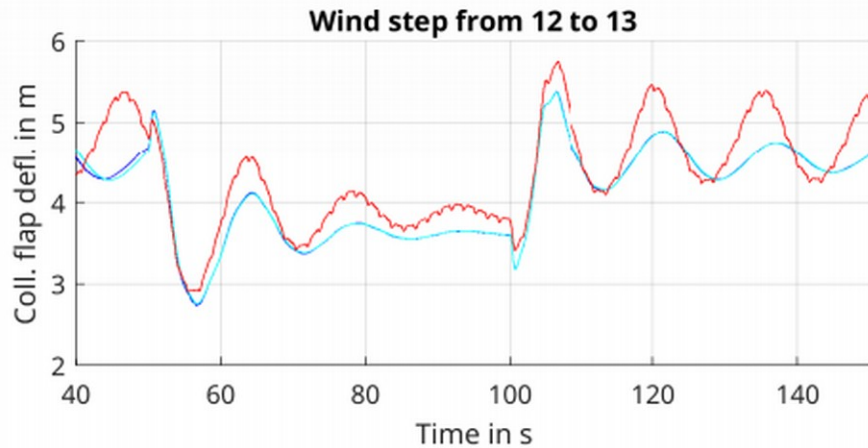
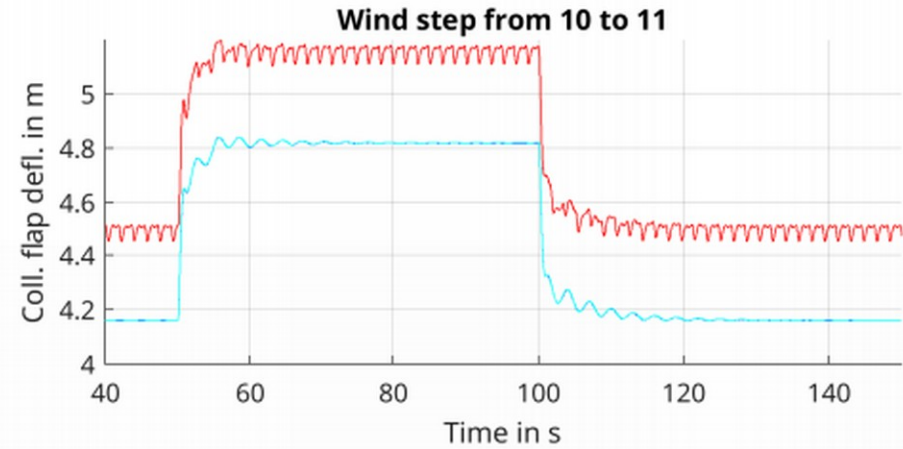
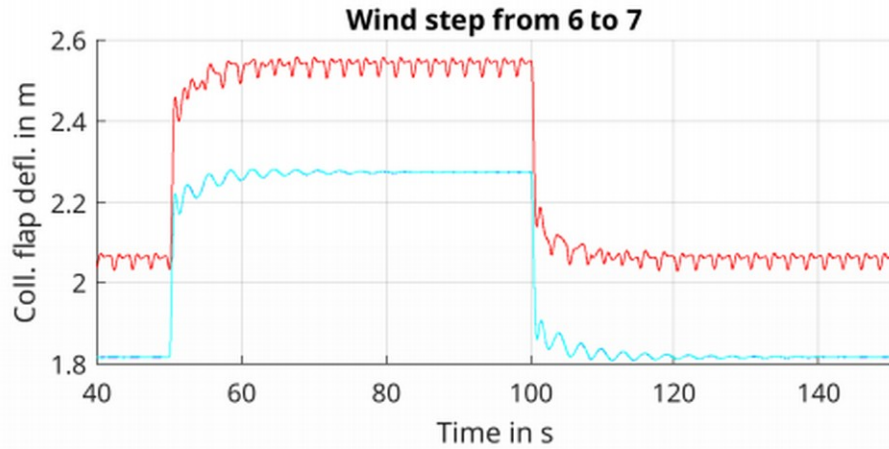
— 6 DOF model — 4 DOF model — 2 DOF model — OpenFAST

# Results – Tower fore-aft Deflection



6 DOF model 4 DOF model 2 DOF model OpenFAST

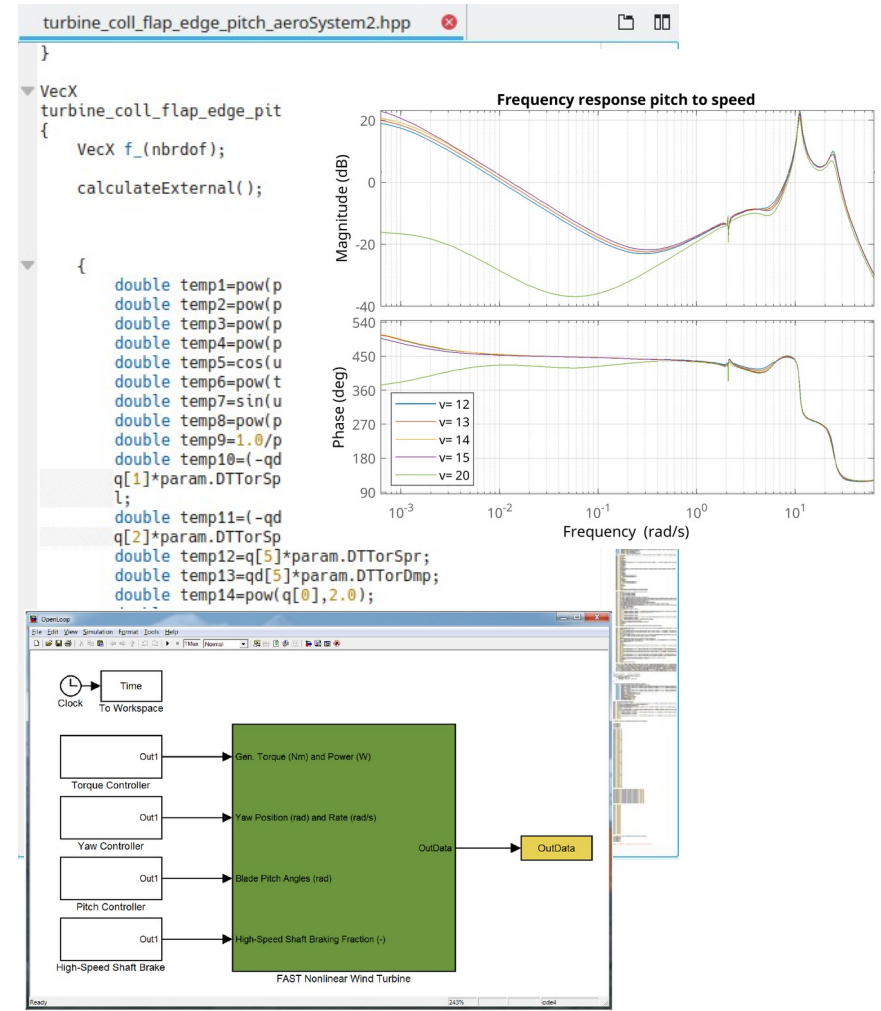
# Results – Collective Blade Flap Deflection



— 6 DOF model — 4 DOF model — OpenFAST

# Applications

- Linear Models (all operating points at once)
  - Classical Controller Design
  - Frequency domain analysis
  - Matrix inequality based control design
- Simulations models, Digital Twins
- Jacobians for advanced observer and controller design
  - Nonlinear Model Predictive Control
  - Extended Kalman Filters
  - Moving Horizon Estimators
- Sensitivities for parameter studies and optimization
- **Generation of custom, high performance code for all these applications**



Jonkmann and Jonkmann 2016

# The Frameworks

- WeLib – Wind Library
  - <https://github.com/ebranlard/welib>
  - Python code
  - Extensive library covering a wide range of aspects
  - Including aero- and hydro-dynamics

Why GitHub? Team Enterprise Explore Marketplace Pricing

Search Sign in Sign up

ebranlard / welib

Code Issues Pull requests Actions Projects Security Insights

master 2 branches 0 tags Go to file Code

data	Update for unittests	2 months ago
welib	Misc updates to cleanup tests	2 months ago
.gitignore	More body wrappers	4 months ago
.travis.yml	Update for unittests	2 months ago
LICENSE.TXT	_small fix of index in MiniBEM	2 years ago
Makefile	Misc updates to cleanup tests	2 months ago
README.md	Misc updates to cleanup tests	2 months ago
requirements.txt	Update for unittests	2 months ago
setup.py	_update of tests	2 years ago

README.md

build passing Donate Buy me a coffee

## welib

Suite of python and matlab tools for aero-servo-hydro-elasticity (aerodynamics, controls, hydrodynamics, structure/elasticity) and wind energy.

About: Wind energy library, matlab and python tools for wind turbines analyses

Releases: No releases published

Packages: No packages published

Languages: Python 68.1%, MATLAB 30.2%, F\* 1.1%, Other 0.6%



# The Frameworks

- CADyn – Computer aided Dynamics
  - <https://github.com/jgeisler0303/CADynTurb>
  - Maxima, MATLAB and C++ code
  - General purpose CADyn MB code plus specific turbine simulator
  - Supporting repositories CCBlade, FEMBeam

The screenshot displays the GitHub repository page for `jgeisler0303/CADynTurb`. The repository is on the `master` branch and contains 20 commits. The file list includes:

File/Folder	Description	Last Update
5MW_Baseline	some fixes; added more outputs to NRELOfshrBslne5MW_Onsh...	23 days ago
matlab	Updated README with more precise instructions	23 days ago
model	some fixes; added more outputs to NRELOfshrBslne5MW_Onsh...	23 days ago
simulator	some fixes; added more outputs to NRELOfshrBslne5MW_Onsh...	23 days ago
.gitignore	restructured MATLAB files for model generation and demonstrati...	3 months ago
LICENSE	Initial commit	12 months ago
README.md	Update README.md	23 days ago

The `README.md` file contains a **Video Tutorial** section with the text: "You can watch a tutorial showing you all of the following steps here: <https://youtu.be/QwgTmSUtoPY>". It also has a **Getting Started** section that begins with: "You need to install Maxima from here: <https://sourceforge.net/projects/maxima/files/>. On Linux make sure you select the Steel Bank Common Lisp version (it has sbcl in its name). Windows you may also choose to install GNU Maxima from here: <https://sourceforge.net/projects/gnumaxima/files/> this will install the

**Thank You for  
Your Interest!**

Prof. Dr.-Ing. Jens Geisler  
May 16, 2021

