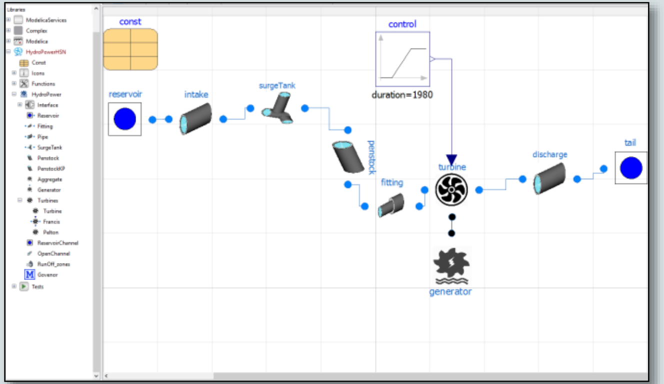
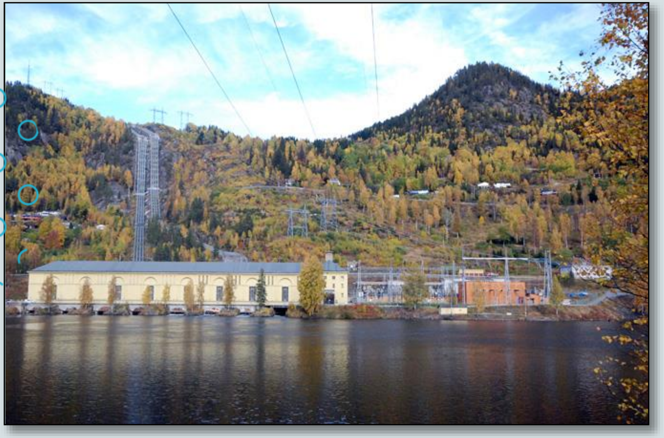
University College of Southeast Norway

TMCC
Telemark Modeling and Control Center



Presentation of Modelica HydroPowerUSN library



PhD: Liubomyr Vytvytskyi
Supr.: Bernt Lie

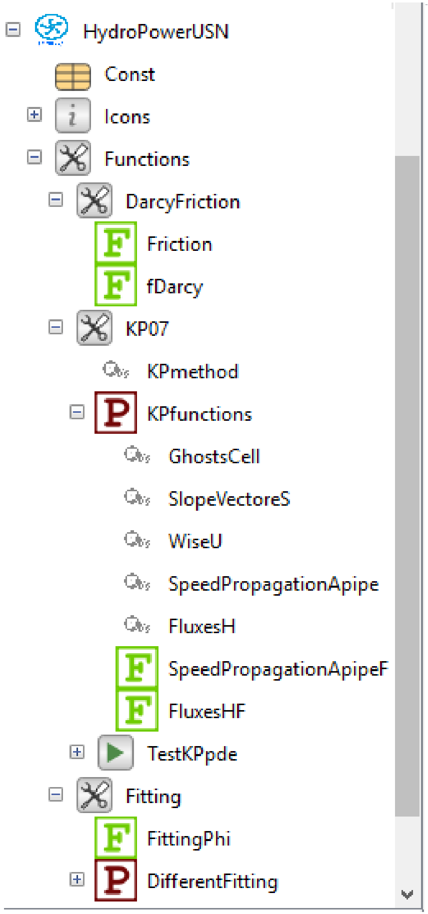
Outline

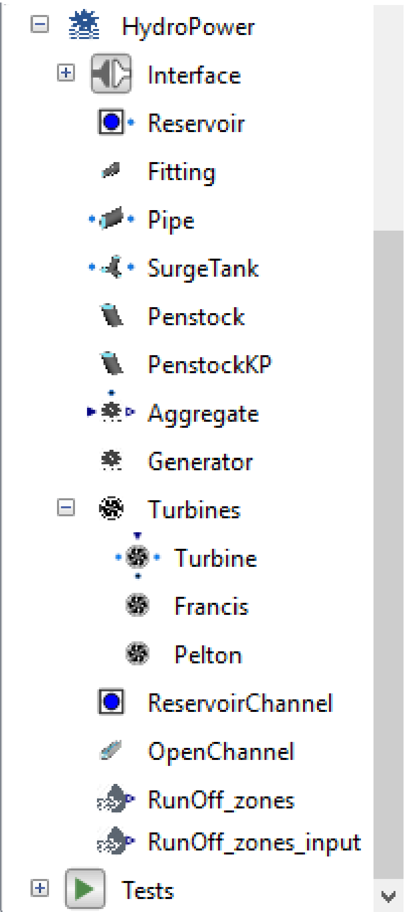
- Introduction
- Waterway modeling
  - Penstock complexity: compressibility & elasticity
  - PDE solver
- Francis turbine modeling
  - Mechanistic model
  - Design algorithm
- Hydrology model
  - Calibration
- Synergy with other Modelica library
- Future work

Introduction

PhD project: «Dynamics and control of integrated energy systems»:

- Hydro power Modelica library in OpenModelica
- Tools for analysis using Python (PythonAPI)
- Possibilities for power flow modelling and control
- At least one case study

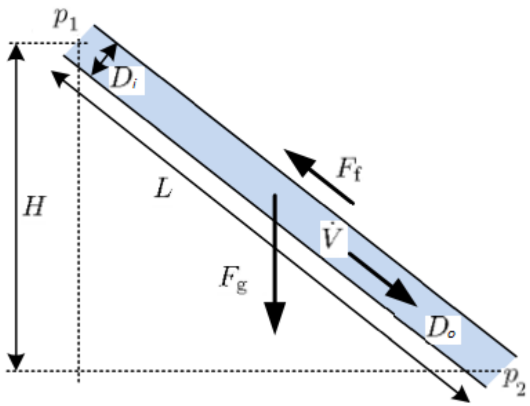




Results so far

- Functions for friction term and pressure drop in different fittings
- Solver for hyperbolic PDE in OpenModelica (Kurganov-Petrova scheme)
- Models for different units of hydropower system
  - Waterway: pipe/conduit, surge tank, penstock (elastic walls with compressible water), etc.
  - Turbines: Francis + design alg.; Pelton
  - Generator
- Hydrology model
- Using Python API

Simple model for waterway pipe



- Mass balance: $\frac{dm}{dt} = \dot{m}_{in} - \dot{m}_{out} = 0$
  - $m = \rho V = \rho L\bar{A};$
  - $\dot{m}_{in} = \rho\dot{V}_{in},\ \dot{m}_{out} = \rho\dot{V}_{out}.$
- Momentum balance: $\frac{dM}{dt} = \dot{M}_{in} - \dot{M}_{out} + F_p + F_g + F_f$
  - $M = mv, \dot{M}_{in} = \dot{m}_{in}v_{in}, \dot{M}_{out} = \dot{m}_{out}v_{out}$
  - $v_{in} = {}^{\dot{V}_{in}}/_{A_{in}},\ v_{out} = {}^{\dot{V}_{out}}/_{A_{out}},\ v = {}^{\dot{V}}/_{\bar{A}}$
  - $F_p = A_{in}p_1 - A_{out}p_2$
  - $F_g = mg\cos\theta$
  - $F_f = -\frac{1}{8}Lf_D\pi\rho\bar{D}v|v|$
- Compressibility coefficients
  - $\beta_T = \frac{1}{\rho}\frac{d\rho}{dp}$
  - $\beta^{eq} = \frac{1}{A}\frac{dA}{dp}$
  - $\beta^{tot} = \beta^{eq} + \beta_T$

Discretization: KP scheme

- PDEs in vectors form:

$$\frac{\partial U}{\partial t} + \frac{\partial F}{\partial x} = S$$

where:

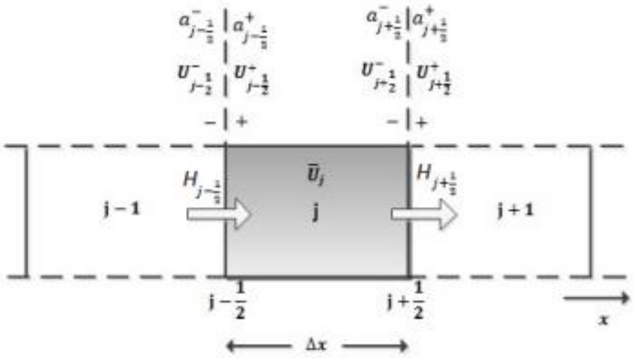
$$U = [p \quad \dot{m}]^T$$

$$F = \left[\frac{\dot{m}}{A^{atm}\rho^{atm}\beta^{tot}} \quad \dot{m}v + Ap\right]^T$$

$$S = \left[0 \quad \rho A g \cos\theta_p - \frac{1}{8} f_D \pi \rho D v|v|\right]^T$$

- Solution:

$$\frac{d}{dt}\bar{U}_i(t) = -\frac{H_{i+\frac{1}{2}}(t) - H_{i-\frac{1}{2}}(t)}{\Delta x} + \bar{S}_i(t)$$



- The central upwind numerical fluxes:

$$H_{i+\frac{1}{2}}(t) = \frac{a^+_{i+\frac{1}{2}} F\left(U^-_{i+\frac{1}{2}}\right) - a^-_{i+\frac{1}{2}} F\left(U^+_{i+\frac{1}{2}}\right)}{a^+_{i+\frac{1}{2}} - a^-_{i+\frac{1}{2}}} + \frac{a^+_{i+\frac{1}{2}} a^-_{i+\frac{1}{2}}}{a^+_{i+\frac{1}{2}} - a^-_{i+\frac{1}{2}}}\left[U^+_{i+\frac{1}{2}} - U^-_{i+\frac{1}{2}}\right]$$

$$H_{i-\frac{1}{2}}(t) = \frac{a^+_{i-\frac{1}{2}} F\left(U^-_{i-\frac{1}{2}}\right) - a^-_{i-\frac{1}{2}} F\left(U^+_{i-\frac{1}{2}}\right)}{a^+_{i-\frac{1}{2}} - a^-_{i-\frac{1}{2}}} + \frac{a^+_{i-\frac{1}{2}} a^-_{i-\frac{1}{2}}}{a^+_{i-\frac{1}{2}} - a^-_{i-\frac{1}{2}}}\left[U^+_{i-\frac{1}{2}} - U^-_{i-\frac{1}{2}}\right]$$

- The one-sided local speed of propagations:

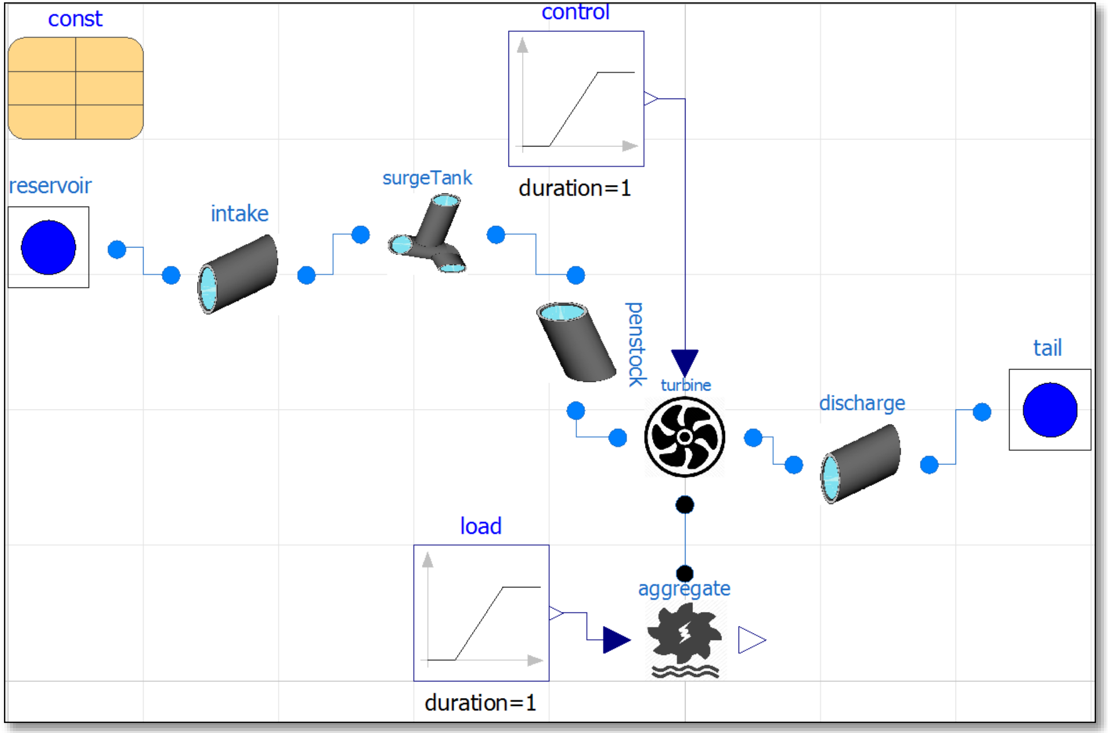
$$a^+_{i\pm\frac{1}{2}} = max\left(\lambda^+_{1,i\pm\frac{1}{2}}, \lambda^-_{1,i\pm\frac{1}{2}}, 0\right)$$

$$a^-_{i\pm\frac{1}{2}} = min\left(\lambda^+_{2,i\pm\frac{1}{2}}, \lambda^-_{2,i\pm\frac{1}{2}}, 0\right)$$

- The eigenvalues:

$$\lambda_{1,2} = \frac{v \pm \sqrt{v^2 - \frac{4A}{A^{atm}\rho^{atm}\beta^{tot}}}}{2}$$

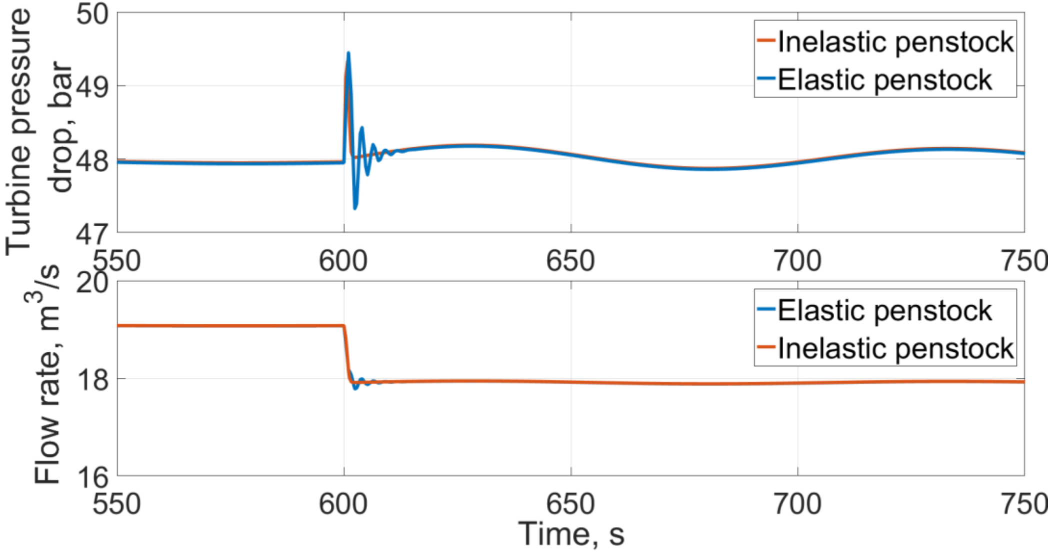
Simulation: Sundsbarm hydropower plant



- Difference in:
    - Oscillations
        - Inelastic penstock – smoother
    - Simulation time
        - Elastic penstock – 3 times longer

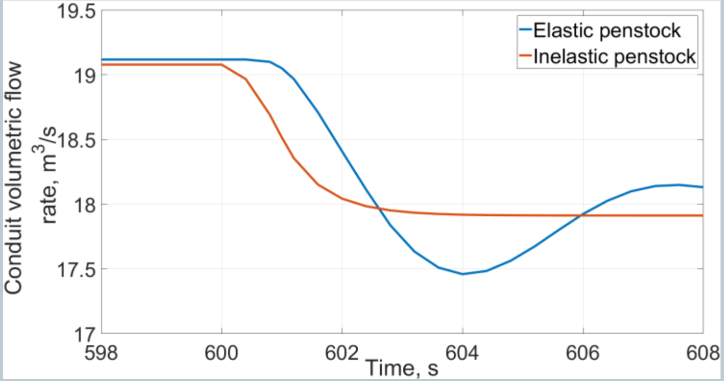
- Why?
    - More complexity
        - Better dynamics
        - Slower simulations

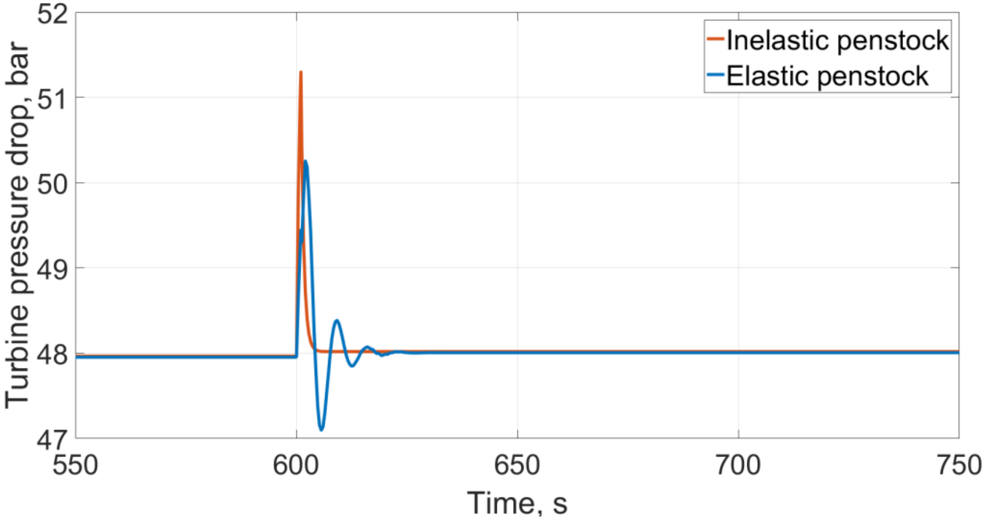
First simulation scenario



Second simulation scenario (without surge tank)

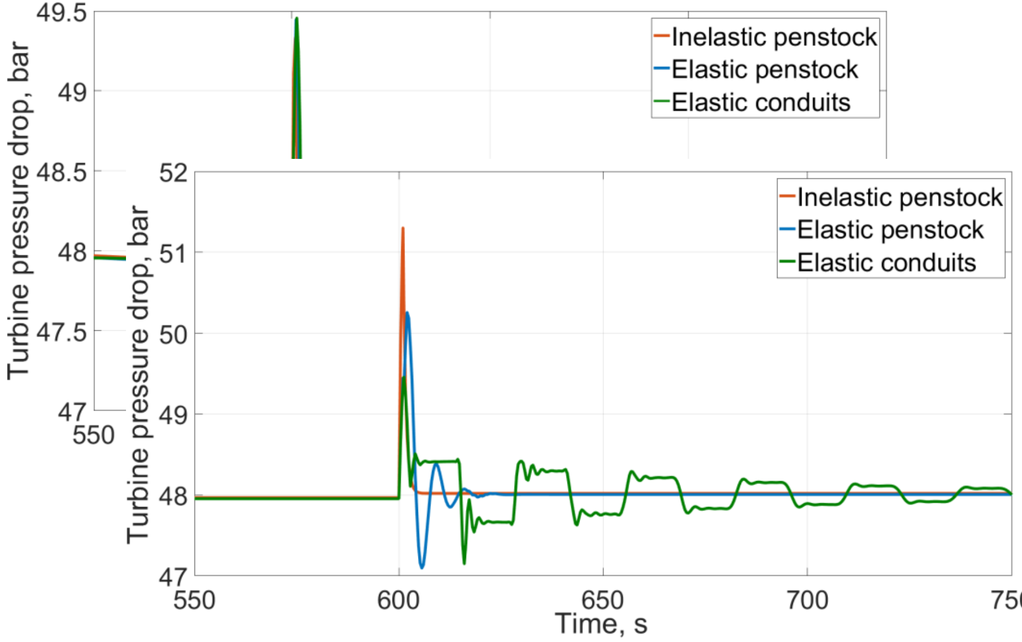
- Why does amplitude differ?
  - Simple penstock model
    - Volumetric flow rate changes simultaneously through the whole system
  - Penstock model with elastic walls
    - Not simultaneous due to wave propagation





- Simulation time increases dramatic – ca. 20-30 times.
- More time to reach a new steady state
  - caused by the water wave moving back and forth through the whole system.
  - one of the reasons of using the surge tank

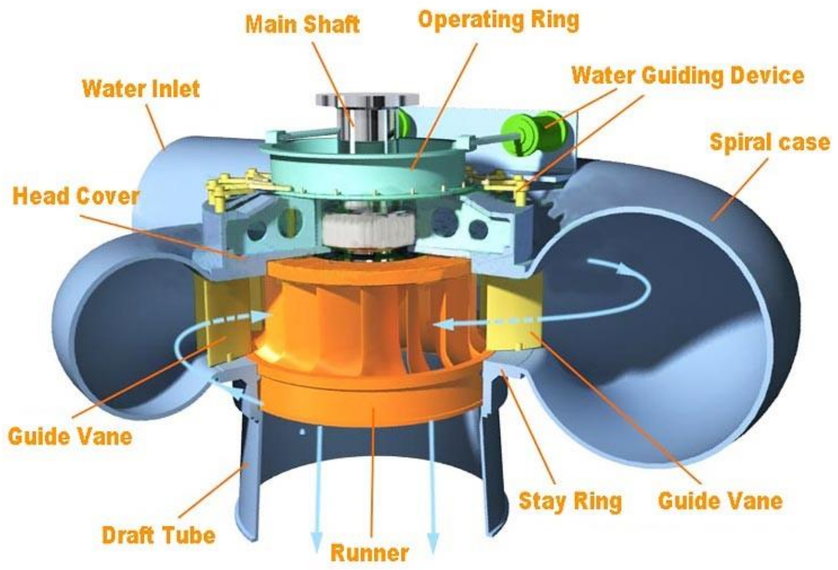
Third simulation scenario (elastic conduit + penstock)



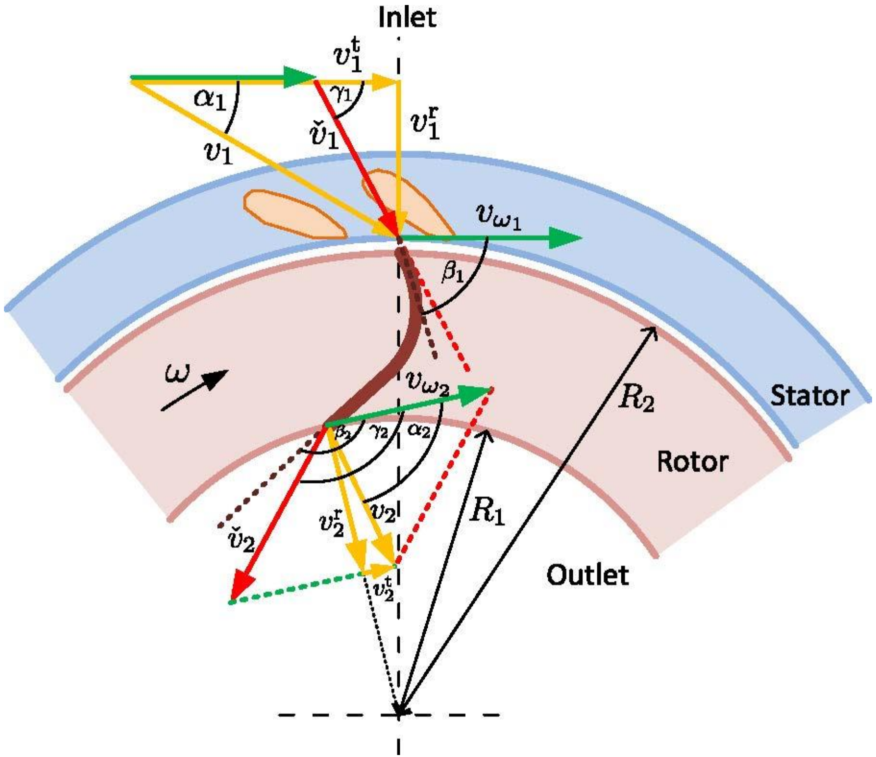
Francis turbine

Water flows through:

- Spiral case
  - Distribute the flow rate approx. the same for guide vanes.
- Guide vanes
  - Steer the water at a certain angel towards the runner
- Runner with runner blades
  - Leads to turbine rotation and a pressure drop
- Draft tube
  - Increase the outlet turbine pressure



Mechanistic Francis turbine model



- Francis turbine shaft power:

$$\dot{W}_s = \dot{m}\omega\left(R_1\frac{\dot{V}}{A_1}\cot\alpha_1 - R_2\left(\omega R_2 + \frac{\dot{V}}{A_2}\cot\beta_2\right)\right)$$

- Turbine total work rate:

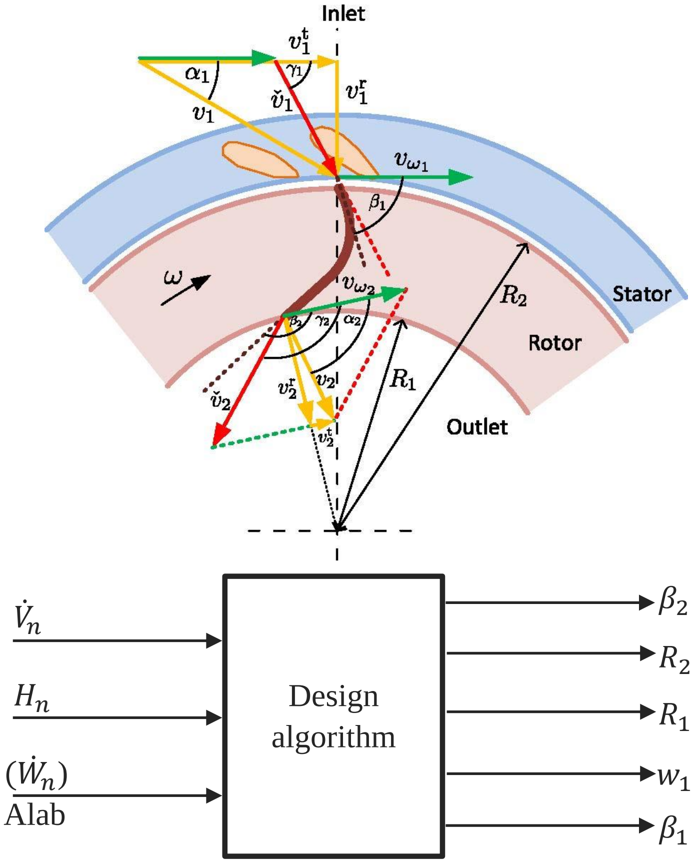
$$\dot{W}_t = \dot{W}_s + \dot{W}_{ft} + \Delta p_{\mathrm{V}}\dot{V}$$

- Turbine friction term:

$$\dot{W}_{ft} = k_{ft,1}\dot{V}(\cot\gamma_1 - \cot\beta_1)^2 + k_{ft,2}\dot{V}\cot{\alpha_2}^2 + k_{ft,3}\dot{V}^2$$

- Efficiency of the turbine:

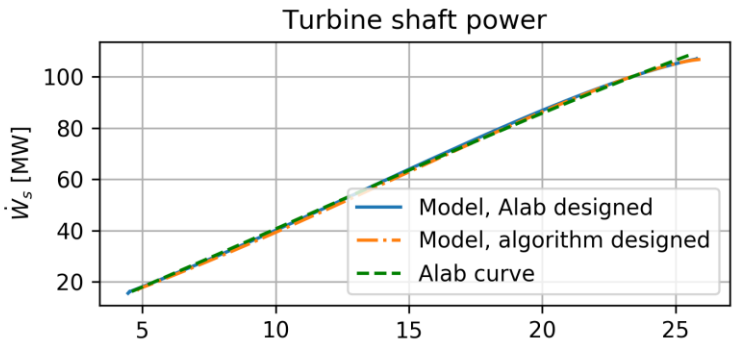
$$\eta = \frac{\dot{W}_s}{\dot{W}_t}$$

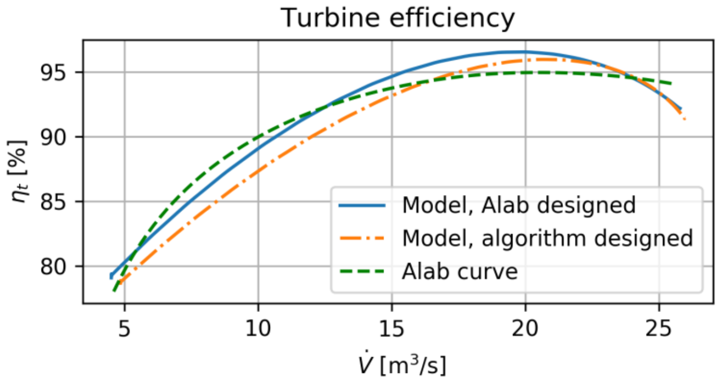


Francis turbine design algorithm

Based on Brekke studies:

1. Choose the outlet blade angel $\beta_2 = 162.5°$ and reference velocity $v_{w,2} = 41m/s$
2. Define the outlet runner cross section area A_2 (radius R_2)
3. Finding the inlet runner dimension:
   - Inlet cross section area A_1 (radius R_1)
   - Inlet runner width w_1
4. Define the inlet blade angel β_1





Design	β_2, [°]	R_2, [m]	R_1, [m]	w_1, [m]	β_1, [°]
Algorithm	162.5	0.777	1.32	0.25	62.85
Alab	162.4	0.775	1.32	0.2	70.77

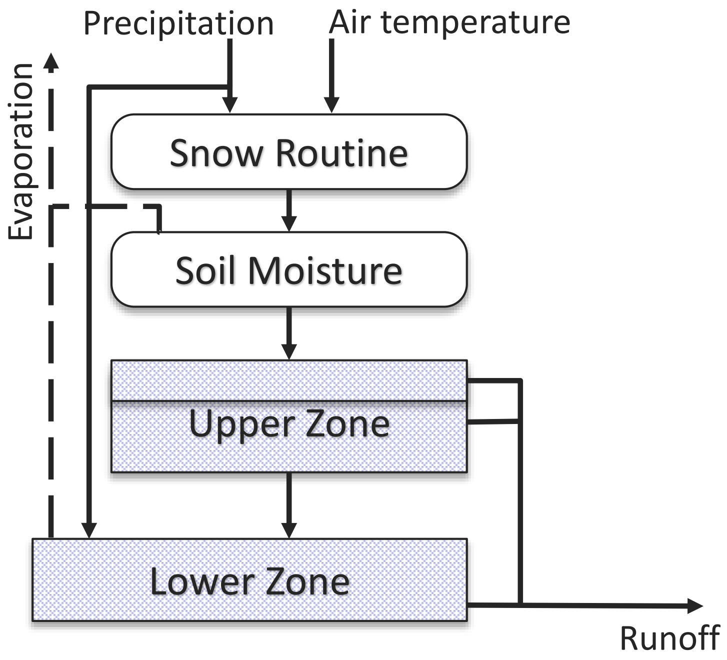
Comparison and fitting

- Mechanistic model fit the *Alab* results quite well
- Deviation in the turbine efficiency caused by differences in calculation
  - *Alab* use nominal static net head
  - Model handle system dynamics
- Design algorithm shows reasonable results in comparison to *Alab*. Difference:
  - *Alab*: $A_2 = 1.1A_1 \Rightarrow v_1^r = 1.1v_2^r$
  - Algorithm: $A_1 = 1.1A_2 \Rightarrow v_2^r = 1.1v_1^r$

Structure:

- Snow routine
  - Snow storage
  - Free water contents in snow
  - Snow melt
- Soil moisture
  - Receive rainfall or snow melt
  - Computes the storage of water
  - Computes actual evapotranspiration
- Runoff response routine transforms the net precipitation into runoff
  - Upper zone – quick runoff components
  - Lower zone – represents the groundwater and lake storage

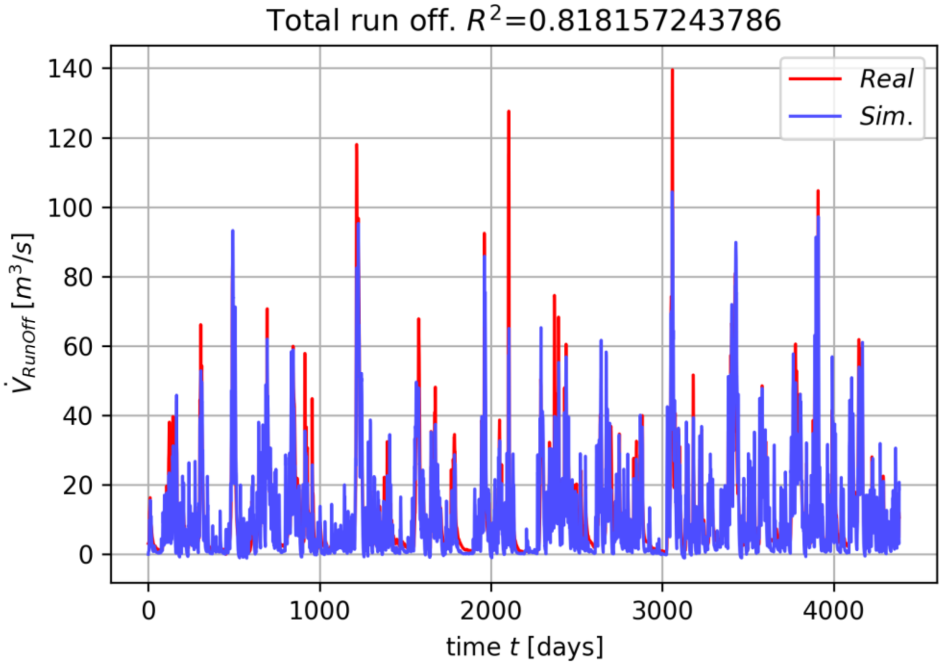
Hydrology model (HBV)



Model calibration:

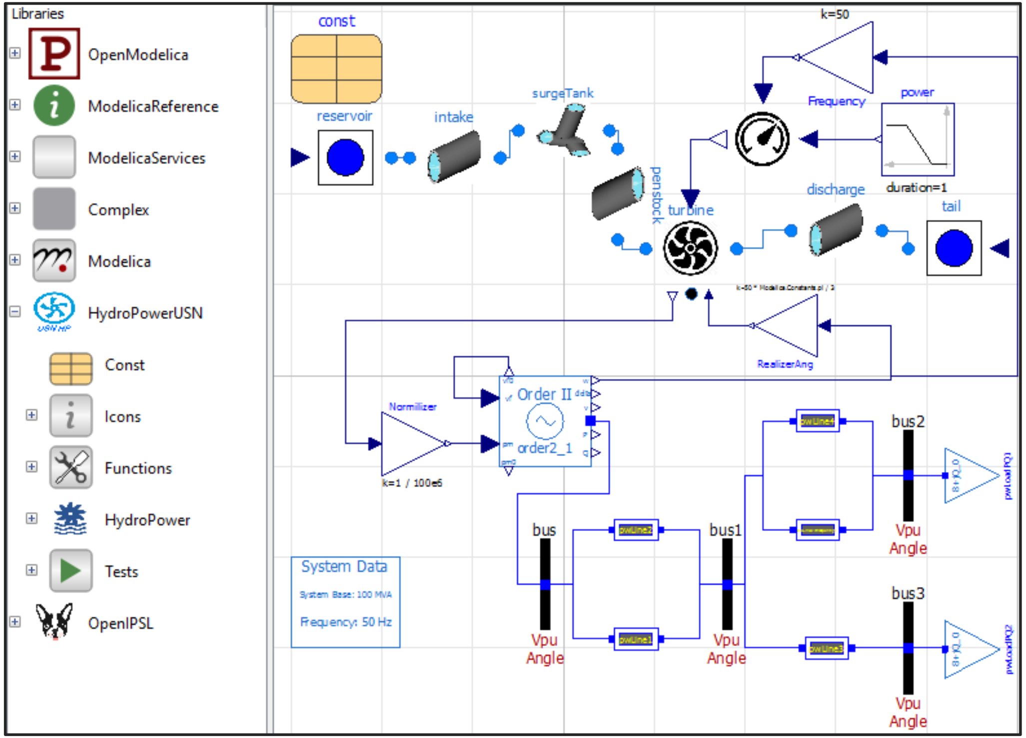
- Threshold temperature
- Degree-day factor
- Precipitation correction – Rainfall
- Precipitation correction– Snowfall
- Field capacity in soil moisture
- β parameter in soil moisture
- Threshold level for quick runoff
- Percolation from upper to lower zone
- Recession constants for upper and lower zones

Hydrology model (HBV)



Synergy with the Modelica library – OpenIPSL

- OpenIPSL – Open-Instance Power System Library
- Much wider possibilities for modelling the power system:
  - Variety in generator model complexity
  - Variety in governor types
  - A lot of other power system components
- Possibility to create a whole hydropower system: from precipitation/reservoir – to electricity consumers



Possibilities for future work

- Complete the library:
  - Waterway part
    - Water temperature variation, close surge tank, etc.
  - Turbines
    - Francis (losses), Pelton (test), Kaplan
  - Electrical part
    - Generator, governor, power grid (transformers, lines, etc.)
  - Multi scaling for different time domain
- Tools for analysis large scale hydropower system using Python (Python API)
- Power flow modelling and control
- Case study

The end!