

Documentation

Matlab Codes

**ExpanderModel.mat**

1. **Model name:**

ExpanderModel.mat

1. **Model description:**

ExpanderModel.mat is a matlab function developed for the simulation of volumetric expanders. Three modelling paradigms are currently implemented to simulate the expander, i.e.

* CstEff: a “constant efficiency” model into which constant values for both the isentropic efficiency and the filling factor are provided by the user, i.e.

Heat losses to the ambience are also included by means of a global AUloss coefficients i.e.

* PolEff: a “polynomial efficiency” model into which quadratic polynomial regressions are used to evaluate both the volumetric efficiency (i.e. the filling factor) and the isentropic efficiency, i.e.

Heat losses to the ambience are also included by means of a global AUloss coefficients as in the CstEff model.

* SemiEmp: a “semi-empirical” model implementing physics-based equations as proposed by Lemort et al. in [1]. This model decomposes the evolution of the fluid into a series of 6 consecutives steps and a detailed description of the modelling scheme can be found in [1].[[1]](#footnote-1)

1. **Model inputs:**

The model inputs are the following ones:

* P\_su (Pa) : inlet pressure of the pump;
* P\_ex (Pa) : outlet pressure of the pump;
* h\_su (J/kg) : inlet enthalpy of the pump;
* fluid (-) : fluid name;
* N\_exp (rpm) : pump rotational speed;
* T\_amb (K) : ambient temperature;
* param : a structure variable that contains the model parameters (see next section);

1. **Model parameters:**

Depending of the type of model chosen by the user, *param* will need to include the following variables:

* if param.modelType = 'CstEff':
  + param.V\_s (m³) : machine displacement volume;
  + param.V (m³) : machine volume;
  + param.epsilon\_is (-): isentropic efficiency;
  + param.FF (-): filling factor (volumetric efficiency);
  + param.AU (W/K) : global heat losses coefficient;
* if param.modelType = 'PolEff':
  + param.V\_s (m³), machine displacement volume;
  + param.V (m³) : machine volume;
  + param.N\_pp\_nom, pump nominal shaft speed;
  + param.coeffPol\_is (-), polynomial coefficients for epsilon\_is;
  + param.coeffPol\_ff (-), polynomial coefficients for FF;
  + param.AU (W/K) : global heat losses coefficient;
* if param.modelType = 'SemiEmp':
  + param.V\_s (m³), machine displacement volume;
  + param.V (m³) : machine volume;
  + param.alpha (-), proportional losses coefficient;
  + param.W\_dot\_loss\_0 (W), constant losses term;
  + param.C\_loss (Nm), losses torque;
  + param.r\_v\_in (-), built-in volumetric ration;
  + param.A\_leak0 (m²), nozzle cross section area for the leakage;
  + param.d\_su (m), nozzle diameter for supply pressure drop;
  + param.AU\_su\_n (W/K), global heat transfer coefficient for supply heat transfer;
  + param.AU\_ex\_n (W/K), global heat transfer coefficient for exhaust heat transfer;
  + param.AU\_amb (W/K), global heat transfer coefficient for ambient heat losses;
  + param.M\_dot\_n (kg/s), nominal mass flow rate;

1. **Model outputs:**

The outputs are two structure variables, namely *out* and *TS*, and they contain the following sub-variables:

* **out** (structure variable with all the relevant model outputs). At minimum, **out** will include
  + T\_ex (K): outlet temperature of the expander;
  + h\_ex (J/kg): outlet temperature of the expander;
  + M (kg) : mass of fluid inside the component;
  + M\_dot (kg/s) : fluid mass flow rate;
  + W\_dot (W) : net mechanical power;
  + Q\_dot\_amb (W) : heat losses;
  + epsilon\_is (-) : expander isentropic efficiency;
  + FF (-) : filling factor;
  + time (sec) : the simulation time
  + flag (-): the model flag (>0 if the model run correctly, <0 if not)

In the case of the SemiEmp model, **out** will also include all the intrinsic variables of the model;

* **TS** (structure variable only used to plot the Ts diagram)
  + T (K) : vector of the fluid temperatures
  + s (J/kg.K) : vector of the fluid entropies

1. **External function requirements:**

The user must install CoolProp (<http://www.coolprop.org/>) to run ExpanderModel.

1. **Matlab version:**

This code has been developed under Matlab R2015a

1. **Contact:**

For any further information, please contact one of the main developers of ORCmKit:

* Rémi Dickes ([rdickes@ulg.ac.be](mailto:rdickes@ulg.ac.be)) – University of Liège (Belgium)
* Davide Ziviani ([davide.ziviani@ugent.be](mailto:davide.ziviani@ugent.be) ) – Ghent University (Belgium)

1. [1] V. Lemort, S. Quoilin, C. Cuevas, and J. Lebrun, “Testing and modeling a scroll expander integrated into an Organic Rankine Cycle,” *Appl. Therm. Eng.*, vol. 29, no. 14–15, pp. 3094–3102, 2009. [↑](#footnote-ref-1)