ORCmKit: an open-source library for organic Rankine cycle modelling and analysis

Rémi Dickes [a,#], Davide Ziviani [b], Michel de Paepe [b], Martijn van den Broek [b], Sylvain Quoilin [a] and Vincent Lemort [a]

[a] Thermodynamics and Energetics Laboratory - Department of Mechanical and Aerospace Engineering, Faculty of Applied Sciences - University of Liège - Belgium

[b] Department of Flow, Heat and Combustion Mechanics – Faculty of Engineering and Architecture – Ghent University – Belgium

[#] Contact: rdickes@ulg.ac.be

Abstract:
As for many other technologies, modelling and simulation of organic Rankine cycles (ORCs) are crucial for design, optimization and control purposes. However, model development is often time consuming and the scientific community lacks of open-access tools to study ORC systems. For these reasons, researchers from the universities of Liège and Ghent in Belgium gathered their knowledge and created "ORC modelling Kit" (ORCmKit), an open-source library dedicated to the steady-state simulation and analysis of organic Rankine cycles. Both component-level and cycle-level models are provided and different ORC architectures can be simulated. For each of the main component of ORC systems, different models are available with increasing complexity which allows a wide range of modelling possibilities. In order to remain general and accessible to as many people as possible, three widely used programming languages are covered within ORCmKit, i.e. Matlab, Python and EES (Engineering Equation Solver). Besides source codes, ORCmKit also includes calibration tools for empirical and semi-empirical models as well as a complete documentation for ease of use.

Keywords:
ORC, modelling library, open-source, Matlab, Python, EES

1. Introduction

Because of the depletion of fossil fuels and growing environmental issues, the world of energy is undergoing many changes toward increased sustainability. Among many different fields of research, power generation from low-grade temperature heat sources (such as solar, waste heat recovery or geothermal) is gaining interest because of its enormous worldwide power potential. The organic Rankine cycle (ORC) is one of the most suitable thermal engines for converting low-grade heat into mechanical power and the number of publications dedicated to ORC systems is continuously rising for several years (see Fig. 1). Aside of experimental projects, these publications mostly investigate cycle optimizations, optimal working fluid selections and various techno-economic studies applied to specific case studies. A common aspect of all these works is the use of numerical models to predict the ORC performance under various operating conditions.

Many types of models exists, from the simplest to the most complex, each responding to particular requests. A main distinction can be observed between steady-state and dynamics models. While the firsts predict the system performance under static (or quasi-static) equilibrium, the latest account for the dynamics effects (e.g. energy and mass accumulations in the system components) during transient operating conditions. Steady-state models are either used for sizing purpose or off-design simulations. Dynamic models, on the other hand, are preferably employed to implement and assess the benefits of control strategies under real dynamic operating conditions.

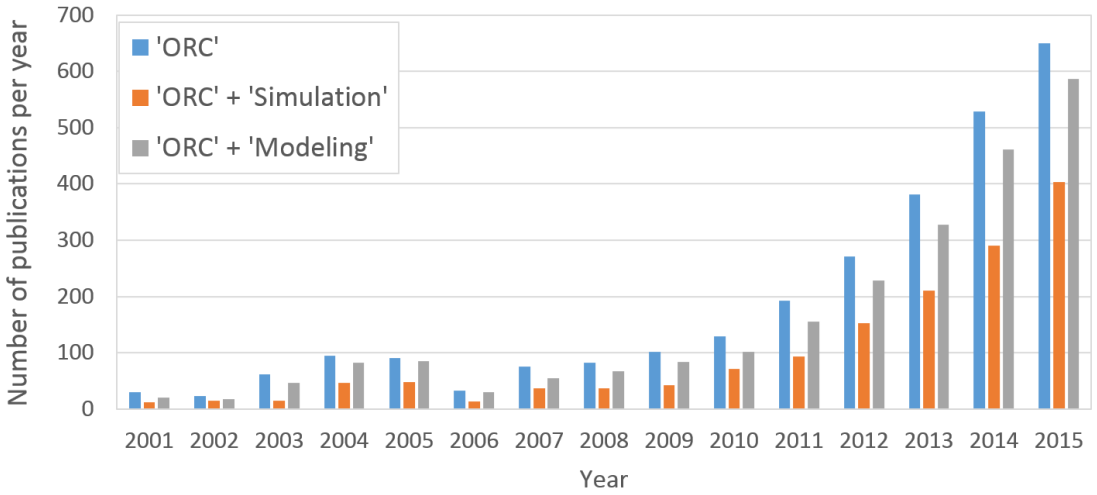

Fig. 1: Number of publications per year related to ORC systems from 2001 to 2015 (source: ScienceDirect)

Commercial modelling platforms exist and one could use SimSci PRO/II [1], Aspen Plus [2], Chemcad [3], AxCycle [4] or Cycle-Tempo [5] for conducting simulations of ORCs (this list is non-exhaustive). However, these commercial tools are generally expensive (up to 10.000$/year) and are not affordable for the academics or small businesses. In most cases, researchers develop their own modelling library in various programming environments (such as Matlab, Python, EES, Excel, Modelica, etc.) and the models are not made available in the public domain. However, model development (including its validation) is time-consuming and the scientific community would greatly benefit of reliable open-source modelling tools. Some open-access simulation packages exist and must be acknowledged. Regarding to dynamic modelling, the University of Liège has developed a Modelica-based library named ThermoCycle and dedicated to the dynamic simulation of thermal systems, including ORCs. Steady-state simulation tools are also available. For example, Poles and Venturin [6] proposed an ORC model in Scilab and Quoilin [7] released several models in EES. Lately, Ziviani et al. ([8], [9]) developed ORCSim, a Python-based simulation software while Dickes et al. [10] developed similar tools in the Matlab environment to perform off-design simulations of ORC systems.

In order to unify their knowledge and to create a single modelling repository, the authors have created 'ORC modelling Kit' (or 'ORCmKit'). The goal of ORCmKit[1] is to provide, at a single place, open-source and reliable models for the steady-state simulation of organic Rankine cycles and their components. Initiated by the authors, ORCmKit is aimed to be useful for the entire scientific community and to be actively updated by researchers in the field. For instance, the repository includes models in three commonly used modelling environment, namely Matlab, Python and EES (Engineering Equation Solver). This paper compares these modelling environments and provides a description of the different models implemented in ORCmKit.

2. Simulation environment

As explained in the introduction, ORCmKit covers for instance three modelling environment, namely Python, Matlab and EES. The next section first compares these environments (a summary is given in Table 1) and then provides additional details regarding the computation of the fluids thermo-physical properties.

[1] ORCmKit library can be accessed on GitHub (https://github.com/orcmkit/ORCmKit) or through KCORC web-site (http://www.kcorc.org/en/open-source-software/)

2.1 – EES

EES [11] (Engineering Equation Solver) is a non-free general equation-solving program that can numerically solve systems of non-linear algebraic and differential equations. It can handle a wide range of technical problems and it is particularly well suited for modelling thermal systems. The programming environment is acausal which permits to reuse the same code even after modifying the models inputs and outputs. For example, a same code can firstly size an ORC system and then be used for assessing the system performance under off-design conditions. Additionally, EES provides many useful functionalities permitting to perform optimizations, uncertainty analysis, units checking or parametric studies. It also contains its own thermodynamic and transport database covering a wide range of substances, including organic fluids and refrigerants. EES is particularly interesting for simple studies and basic thermal systems. It is robust (as long as a proper initialization is ensured by the user) and the model development is really fast. However, simulations with EES demonstrate convergence issues when investigating more complex systems. For example, the part-load simulation of an ORC when only considering the boundary conditions to define the equilibrium state of the system is a real challenge, especially if the ORC features an advanced architecture like several heat exchangers in series (e.g. with a recuperator). Although EES proposes some useful functionalities, it does not provide a user-friendly environment to conduct cascaded tasks.

2.2 – Matlab

Matlab [12] is among the most widely used numerical computing software worldwide. It is non-free but it offers a powerful high-level programming environment applicable for many fields of research, including the modelling of thermal systems. Unlike EES, Matlab has a causal modelling environment and identical codes cannot be used again after switching the model inputs and outputs. Therefore, internal initializations and iterations loop must be self-implemented by the user when modelling implicit systems like closed-loop ORCs. Although time consuming, once a model is properly implemented in Matlab it demonstrates a really good robustness. This robustness is crucial to solve advanced numerical problems (e.g. part-load modelling of an existing ORC module) or to perform iterative processes (e.g. model calibration, control optimization, performance mapping, etc.). Matlab is also more suited for performing cascaded tasks in a single environment (e.g. to perform the following tasks in series: data import, models calibration, multiple simulations and results posttreatment).

2.1 – Python

Python [13] is an open-access and growing alternative to Matlab. It is an object-oriented programming language and it offers similar functionalities and characteristics than Matlab. The Python-based models included in ORCmKit are derived from the simulation tools ORCSim [8]. The models architecture and solution scheme were originally inspired from another open-source software, ACHP [14], dedicated to the simulation of air conditioners and heat pumps. The Python-based models take advantage of the object-oriented environment to achieve high modularity. Therefore, it is quite easy to integrate additional components or models into the overall cycle without impacting the structure of the core of the code. The Python-based library also provides a user friendly GUI (Graphical User Interface) for ease use.

2.4 – Thermo-physical property libraries

When modelling thermal systems such as organic Rankine cycles, an important step is to properly evaluate the thermo-physical properties of the different media involved in the thermal processes. Apart of EES which includes its own thermodynamic and transport database, an external library is generally required to compute the fluids properties. Nowadays, the most widely used library is REFPROP developed by the NIST [15], but it is non-free. In ORCmKit, the open-source CoolProp library [16] is used as the main source to retrieve thermos-physical properties for the wide range of refrigerants and incompressible fluids applicable for ORC applications.

	EES	*Matlab*	*Python*
Distribution	Non-free	Non-free	Open-access
Causality	Acausal	Causal	Causal
Thermophysical library	• EES library • CoolProp • Refprop • user-implemented • etc.	• CoolProp • Refprop • user-implemented • etc.	• CoolProp • Refprop • user-implemented • etc.
Model development	Fast	Slow if complex implicit model	Slow if complex implicit model
Model flexibility	High	Low	Medium (causal but object-oriented modelling)
Model robustness	Low for complex systems	High if properly implemented	High if properly implemented
Cascaded tasks	Not user-friendly	User-friendly	User-friendly

Table 1: Comparison between EES, Matlab and Python for the simulation of ORC systems.

The CoolProp library implements the most accurate equations of state available in the literature, as well as highly efficient tabular interpolation methods to speed up property calculations. It can also be used as an interface layer around to REFPROP which makes it extremely flexible. An external library of lubricant oils has also been integrated into CoolProp which allows to account for the lubricant effects inside oil-flooded or injected expanders, e.g. screw-expanders.

3. Tools description

An organic Rankine cycle is a thermal power system into which an organic working fluid undergoes several processes aiming to convert low-grade heat energy into mechanical work. In order to form the closed-loop cycle, several components are connected in series and they include at minimum a pump, an expansion device (expander or turbine) and two heat exchangers (a condenser and an evaporator). Each of these components can be modelled with numerical methods of various complexities. In order to remain general and accessible to as many people as possible, the ORCmKit library proposes multiple models for each component and covers various ORC architectures. The following section describes these modelling tools as well as additional features of the library.

3.1 – Pump models

A pump is required to move and pressurize the working fluid. The power consumption and the volumetric flow rate depend mainly of the pump rotational speed and the cycle pressure difference. The following modelling methods are implemented in the ORCmKit library to evaluate the pump performance:

- **Efficiency-based model**
  A first method is to characterize the pump with both its isentropic and volumetric efficiencies i.e.

$$\varepsilon_{is,pp} = \frac{\dot{m}_{pp}(h_{ex,is,pp} - h_{su,pp})}{\dot{W}_{pp}}, \qquad (1)$$

$$\varepsilon_{vol,pp} = \frac{\dot{m}_{pp}/\rho_{su,pp}}{V_{swept,pp} N_{pp}}, \qquad (2)$$

  where $\dot{m}_{pp}$, $\dot{W}_{pp}$, N_{pp}, $V_{swept,pp}$ are respectively the fluid mass flow rate, the pump mechanical power, the rotational speed and the pump displacement volume. The efficiencies are directly imposed to constant values by the user or they can be implicitly calculated by means of user-defined correlations (e.g. empirical laws, polynomial regressions, etc.). These correlations permit to account for the effects of the operating conditions on the pump performance. The most relevant variables impacting the pump efficiencies are the rotational speed and the cycle pressure difference. Once the pump efficiencies are known, the pump mass flow rate and the power consumption are easily derived from (1) and (2).

- **Semi-empirical model:**
  Another method for simulating the pump is a semi-empirical model which implement physics-based equations. First, the pump mass flow rate is calculated as an ideal mass flow rate to which an internal recirculation flow is deduced. The mass flow characterizing this leakages are modeled as an incompressible flow through an equivalent orifice. Therefore, the effective flow delivered by the pump is given by

$$\dot{m}_{pp} = \left(\rho_{su,pp} N_{pp} V_{swept,pp}\right) - \left(A_{lk}\sqrt{2\rho_{su,pp}\Delta P_{pp}}\right), \qquad (3)$$

  where ΔP_{pp} is the pressure difference between the inlet and the exhaust of the pump, $\rho_{su,pp}$ is the inlet density of the fluid and A_{lk} is the surface area of the equivalent orifice. The mechanical consumption of the pump is calculated by adding mechanical losses to the isentropic power. These mechanical losses are calculated by means of constant losses $\dot{W}_0$ added to a term proportional to the isentropic power i.e.

$$\dot{W}_{pp} = \dot{W}_0 + (1 + K_0)\left[(\dot{m}_{pp}/\rho_{su,pp})\Delta P_{pp}\right], \qquad (4)$$

  where K_0 is the proportional coefficient of the losses.

As an example, the performance prediction of a gear pump with three different models is shown in Fig. 2.

3.2 – Heat exchanger models

Heat exchangers (HEX) are used to transfer heat power from a hot source to a cold sink. In an ORC, these heat exchangers are used to vaporize and condensate the working fluid as well as to permit internal heat regeneration. The models implemented in the ORCmKit library assume a counter-flow arrangement. In order to model the heat transfer, the following numerical methods are implemented:

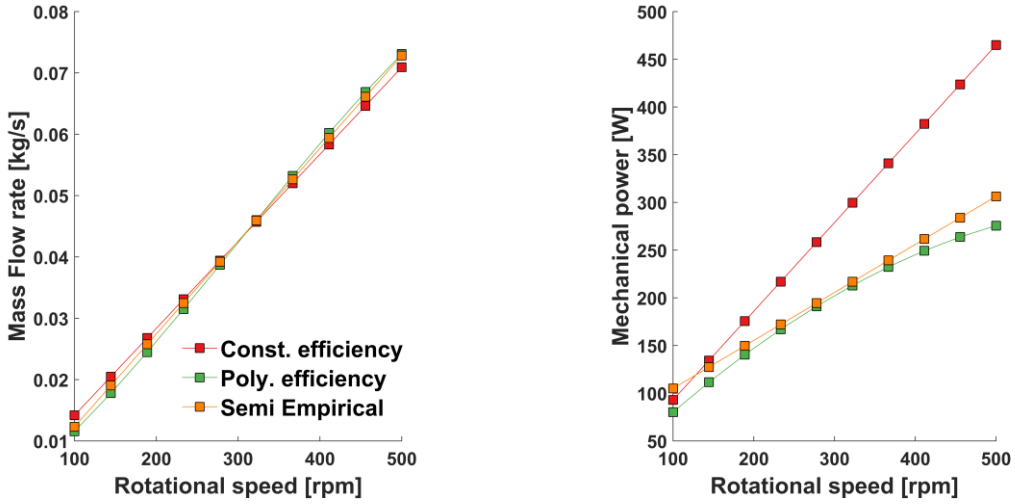

Fig. 2: Example of performance prediction fort a gear pump with three different modeling methods i.e. a constant-efficiency, a polynomial efficiency and a semi-empirical model (simulations with Matlab)

- **Pinch-based model:**
  A first method is to impose the pinch between the temperature profiles of the cold and the hot fluid. The pinch location varies in function of the operating conditions (see Fig. 3) and it is auto-defined within the model. Like for the pump efficiencies, the pinch is either imposed to a constant value or evaluated implicitly by means of a user-defined correlation.

- **Efficiency-based model:**
  The maximum amount of heat power transferrable between two media is the one leading to a pinch equal to zero. In practice, the effective heat power transferred by a heat exchanger is always a fraction of this maximum heat power. A second modelling method is to impose the heat exchanger thermal efficiency ε_{HEX} i.e.

$$\varepsilon_{HEX} = \frac{\dot{Q}_{HEX}}{\dot{Q}_{HEX,\max}}, \qquad (5)$$

  with a constant value or with an empirical correlation specified by the user.

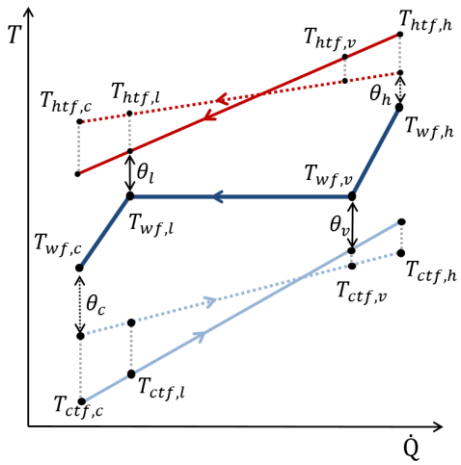

Fig. 3: Pinch location in a three-zone HEX

- **Moving-boundary model:**
  Another numerical method is to decompose the modelling into the different zones of the heat exchanger. In the case of subcritical conditions, the number of zones is between one (neither of the two fluids experience a phase-change) and six (both fluids change from one phase to the other). Each zone transfers a given amount of heat power i.e.

$$\dot{Q}_i = A_i U_i \Delta T_{\log i}\,, \qquad (6)$$

$$U_i = \left(1/h_{conv,h,i} + 1/h_{conv,c,i} \right)^{-1}, \qquad (7)$$

  where A_i is the surface area, U_i is the global heat transfer coefficient and $\Delta T_{\log i}$ is zone logarithmic mean temperature difference. The convective heat transfer coefficients in (6) are either constant, mass flow dependent, or computed by means of Nusselt-based correlations. The effective heat transfer occurring in the heat exchanger is calculated such as the total surface area occupied by the different zones corresponds to the geometrical surface area of the component i.e.

$$\sum_{i=1}^{N} A_i = A_{hex}\,, \qquad (8)$$

  The problem being implicit, EES performs automatically the iterations to compute the effective heat transfer. In Matlab and Python, however, the generalized moving-boundary algorithm proposed by Bell et al. [17] is implemented. The algorithm is further improved to allow for incompressible fluids as heat source or cold sink. Finally, the model can also handle heat exchangers with unequal surface area for the hot and cold side, like finned-tube heat exchangers. For such case, the fin efficiency is computed by means of the Schmidt's method [18].

Two examples of temperature profiles simulated by these HEX models are given in Fig. 4.

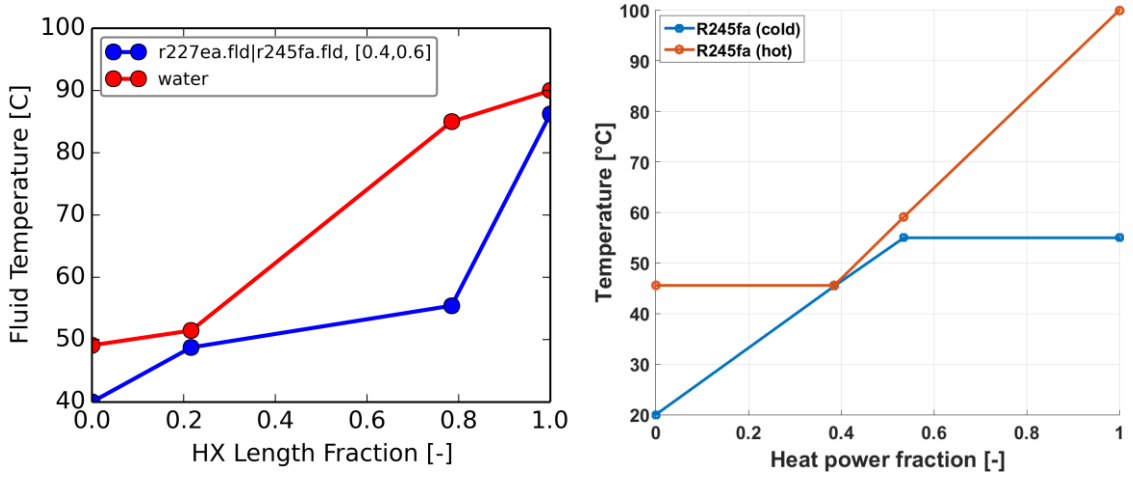

Fig. 4: Temperature profiles in a heat exchanger simulated with the ORCmKit models (left: case of an evaporator with a zeotropic mixture simulated in Python; right: case of a recuperator simulated with R245fa in Matlab)

3.3 – Expander models

An expander is a component used to convert the working fluid energy into mechanical power by means of an expansion process. There are multiple types of expanders, including volumetric and turbomachinery machines, and several modelling methods can be developed to characterize their performance. Therefore, the following numerical models are available in ORCmKit for simulating volumetric expanders:

- **Efficiency-based model:**
  Like for the pump, the expander performance can be evaluated by means of the isentropic efficiency and the filling factor i.e.

$$\varepsilon_{is,\exp} = \frac{\dot{W}_{\exp}}{\dot{m}_{\exp}(h_{su,\exp} - h_{ex,is,\exp})}, \qquad (9)$$

$$FF_{\exp} = \frac{\dot{m}_{\exp} / \rho_{su,\exp}}{V_{swept,\exp} N_{\exp}}, \qquad (10)$$

  where $\dot{m}_{\exp}$, $\dot{W}_{\exp}$, $N_{\exp}$ and $V_{swept,\exp}$ are respectively the fluid mass flow rate, the expander mechanical power, the rotational speed and the expander displacement volume. Once again, these efficiencies can either be directly imposed by the user or implicitly calculated by means of user-defined correlations. Both quadratic functions and advanced empirical correlations (e.g. Pacejka's law [19], see Fig. 5) are implemented to account for the effects of the operating conditions on the expander performance. Moreover, the working temperature in an expander is higher than in a pump resulting in unnegligible heat losses if the expander is not properly insulated. In order to account for the effect of these losses on the exhaust conditions, a third parameter AU_{loss} can be added to the model i.e.:

$$\dot{m}_{\exp}(h_{su,\exp} - h_{ex,\exp}) = \dot{W}_{\exp} + AU_{loss}(\overline{T}_{\exp} - T_{amb}), \qquad (11)$$

- **Semi-empirical model:**
  The second approach chosen for simulating the expander performance is the semi-empirical model proposed by Lemort [20]. The conceptual scheme of the model is depicted in Fig. 6 and a complete description of the governing equations can be found in [21]. The expansion process is decomposed in several steps in order to accounts for mechanical losses, internal leakages, pressure drops, heat losses and under-/over-expansion losses. The semi-empirical model can extrapolate the expander performance in a wide range of operating conditions by the single identification of eight parameters.

3.5 – Pipeline models

Pressure drops and heat losses occurring along the pipes can also be simulated. Heat losses are evaluated by means of an overall heat transfer coefficient i.e. $\dot{Q}_{loss} = AU_{loss}(T_{wf} - T_{amb})$ while pressure losses are obtained as a linear function of the fluid kinetic energy ($\varphi = \dot{m}^2 / \rho$).

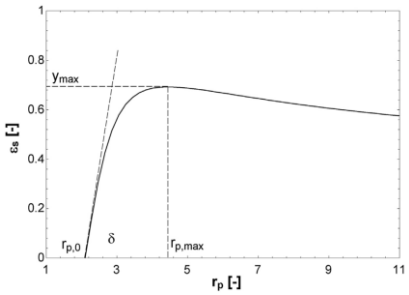

Fig. 5: Pacejka's correlation for estimating the expander isentropic efficiency [19]

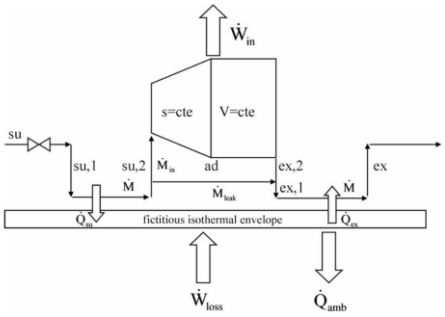

Fig. 6: Semi-empirical model of a volumetric expander [21]

3.4 – ORC models

As mentioned in the introduction, ORCmKit provides both component-level and cycle-level modelling tools. As depicted in Fig. 7, various cycle architectures can be simulated, including recuperative and non-recuperative ORCs, cycles with several heat exchangers in series (e.g. a preheater, an evaporator and a superheater in the high pressure line), ORCs with liquid-flooded expansion, etc. Examples of ORC simulations are illustrated in Fig. 8 for different scenarios. The ORC model is obtained by coupling in series the different components and the user can choose the modelling approach for characterizing each of the ORC subsystems, resulting in very a versatile modelling tool. Since the overall cycle model cannot be expressed causally, implicit iterations are required to get the system steady-state working conditions. Unlike EES which can handle acausal models, both Matlab and Python models implement a robust solver pre-conditioner and overall cycle solver. The preconditioner aim at defining proper initial conditions while the cycle solver drives several residuals to zero by means of a multi-variable algorithm. Further details regarding the solution scheme and the solver algorithm are given in the ORCmKit documentation (see the next section).

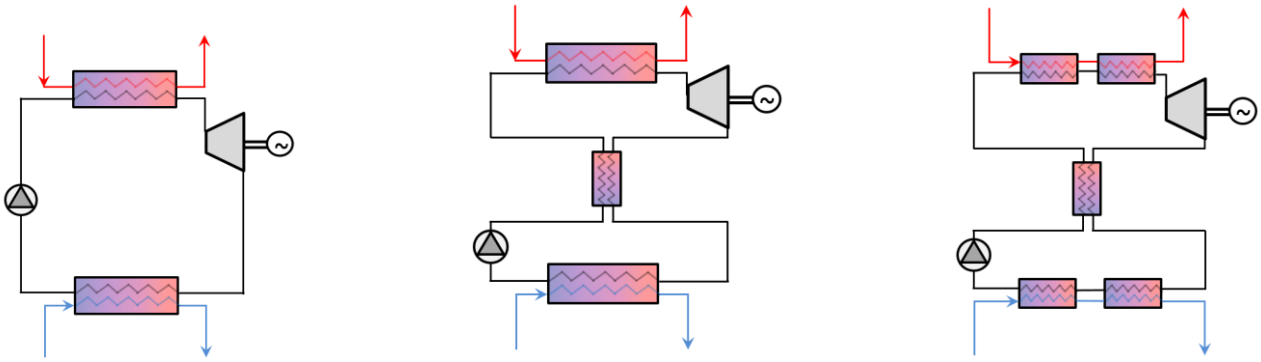

Fig. 7: Examples of ORC configurations covered within the ORCmKit library

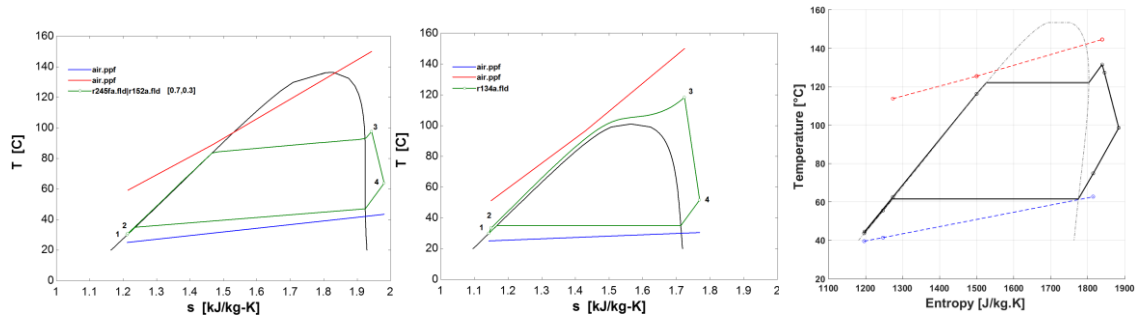

Fig. 8: Examples of ORC simulations performed with Matlab and EES (left: ORC with zeotropic mixture - center: transcritical ORC - right: recuperative ORC with 5 heat exchangers)

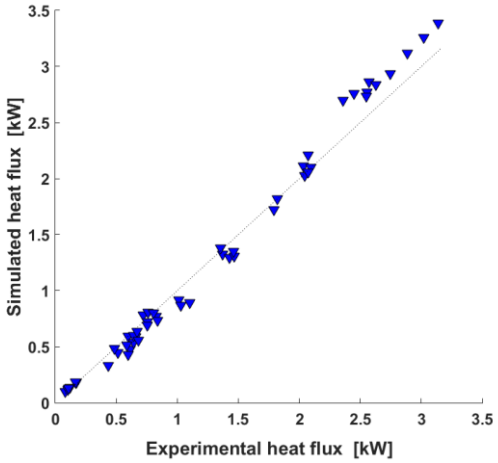

Fig. 9: Parity plot of the heat power transferred in a recuperator after the calibration of a 3-zone moving boundary model

3.5 – Additional features

Additionally to the models described in the previous pages, ORCmKit includes other useful features, i.e.:

- Calibration tools in Matlab and Python are provided to calibrate automatically any models of the library. Based on reference data provided by the user, these calibration codes optimize the model parameters so as to minimize residuals on the model output. Parity plots after the calibration of a recuperator and a scroll expander are given as examples in Fig. 9 and 10 respectively.

- As illustrated along this paper, models in ORCmKit also provides convenient pre-implemented graphical tools to depict the system performance at the component-level (e.g. plot of the temperature profiles in the heat exchangers, see Fig. 4) and at the cycle-level (e.g. T-s diagram and P-h diagram of the ORC, see Fig. 8).

- A complete documentation describing each source code and the different models is provided for ease use.

- A user-friendly GUI (graphical user interface) is also implemented with the Python-based models.

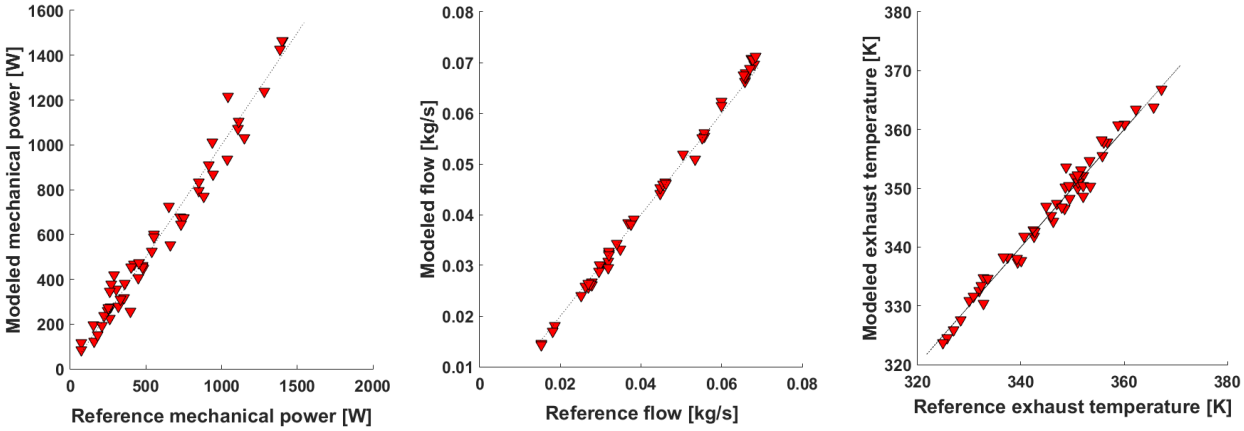

Fig. 10: Parity plot of the mechanical power, the mass flow and the exhaust temperature after calibrating the semi-empirical model applied for a scroll expander.

4. Conclusion

The number of researches related to organic Rankine cycles has drastically increased in the past years. Most of these works involved numerical simulation of ORC systems but open-access modelling libraries are missing. In this context, an open-source library of ORC models is proposed. The library is named ORCmKit and proposes models in three modelling environment, namely EES, Matlab and Python. Both component-level and cycle-level models are implemented. Each component can be simulated with models of various level of complexity as described along this paper. Additionally to these models, ORCmKit includes useful features such calibration codes, graphical tools and a complete documentation. Initiated by researchers of the universities of Liège and Ghent, ORCmKit is aimed to be useful for the entire scientific community and to be actively improved by researchers in the field.

Nomenclature

Acronyms	
EES	Engineering Equation Solver
HEX	Heat Exchanger
HTF	Heat Transfer Fluid
GUI	Graphical User Interface
ORC	Organic Rankine Cycle
ORCLFE	ORC with liquid-flooded expansion
ORCmKit	ORC modelling kit

Symbols	
A	surface area (m²)
FF	filling factor (-)
h	enthalpy (J/kg)
$\dot{m}$	mass flow rate (kg/s)
N	rotational speed (rev/s)
P	pressure (Pa)
$\dot{Q}$	heat power (W)
T	temperature (°C)
U	heat transfer coefficient (W/m².K)
V	volume (m³)
$\dot{W}$	mechanical power (W)
ε	Efficiency, -
ρ	Density, kg/m³

Subscripts	
amb	ambient
c	cold
conv	convection
ex	exhaust
exp	expander
h	hot
is	isentropic
lk	leak
log	logarithmic
max	maximum
pp	pump
su	supply
vol	volumetric
wf	working fluid

References

[1] Schneider Electric, “SimSci PRO / II - Comprehensive Process Simulation (datasheet).” [Online]. Available: http://software.schneider-electric.com/products/simsci/design/pro-ii/.

[2] “AspenTech products - Aspen Plus software.” [Online]. Available: http://www.aspentech.com/.

[3] “Chemcad website.” [Online]. Available: http://www.chemcad.fr/en/.

[4] “SoftInWay products - AxCycle software.” [Online]. Available: http://www.softinway.com/product-and-services/product/axcycle/.

[5] “Asimptote products - Cycle Tempo software.” [Online]. Available: http://www.asimptote.nl/software/cycle-tempo/.

[6] S. Poles and M. Venturin, “Numerical Simulation of an organic Rankine cycle in Scilab.” [Online]. Available: http://www.openeering.com/node/81.

[7] S. Quoilin, “Sustainable Energy Conversion Through the Use of Organic Rankine Cycles for Waste Heat Recovery and Solar Applications,” University of Liège, 2011.

[8] D. Ziviani, B. J. Woodland, E. Georges, E. a. Groll, J. E. Braun, W. T. Horton, M. De Paepe, and M. Van Den Broek, “ORCSIM : A generalised organic Rankine cycle Simulation tool,” in *Proceedings of ASME ORC 2015*, 2015.

[9] D. Ziviani, B. J. Woodland, E. Georges, E. a Groll, J. E. Braun, and W. T. Horton, “Development of a general Organic Rankine Cycle simulation tool : ORCSim,” in *Proceedings of ECOS 2015*, 2015.

[10] R. Dickes, O. Dumont, A. Legros, S. Quoilin, and V. Lemort, “Analysis and comparison of different modeling approaches for the simulation of a micro-scale organic Rankine cycle power plant,” in *Proceedings of ASME -ORC 2015*, 2015.

[11] “FChart software - EES.” [Online]. Available: http://www.fchart.com/ees/.

[12] “MathWorks products - Matlab software.” [Online]. Available: http://nl.mathworks.com/products/matlab/.

[13] “Python official website.” [Online]. Available: https://www.python.org/.

[14] “ACHP website.” [Online]. Available: http://achp.sourceforge.net/.

[15] “NIST products - REFPROP website.” [Online]. Available: http://www.nist.gov/srd/nist23.cfm.

[16] I. H. Bell, J. Wronski, S. Quoilin, and V. Lemort, “Pure and pseudo-pure fluid thermophysical property evaluation and the open-source thermophysical property library coolprop,” *Ind. Eng. Chem. Res.*, vol. 53, no. 6, pp. 2498–2508, 2014.

[17] I. H. Bell, S. Quoilin, E. Georges, J. E. Braun, E. A. Groll, T. W. Horton, and V. Lemort, “A generalized moving-boundary algorithm to predict the heat transfer rate of counterflow heat exchangers for any phase configuration,” *Appl. Therm. Eng.*, vol. 79, pp. 192–201, 2015.

[18] S. W. Stewart, “Enhanced Finned-Tube Condenser Design and Optimization,” Georgia Institute of Technology, 2003.

[19] S. Declaye, S. Quoilin, L. Guillaume, and V. Lemort, “Experimental study on an open-drive scroll expander integrated into an ORC (Organic Rankine Cycle) system with R245fa as working fluid,” *Energy*, vol. 55, pp. 173–183, 2013.

[20] V. Lemort, “Contribution to the characterization of scroll machines in Compressor and Expander modes,” University of Liège, 2008.

[21] 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.