Fast Calculation of Refrigerant Properties in Vapor Compression Cycles Using Spline-Based Table Look-Up Method (SBTL)

Lixiang Li[1] Jesse Gohl[1] John Batteh[1] Christopher Greiner[2] Kai Wang[2]

[1]Modelon Inc, USA, {lixiang.li, jesse.gohl, john.batteh}@modelon.com
[2]Ford Motor Company, USA, {cgreiner, kwang37}@ford.com

Abstract

Refrigerant property calculation has a significant impact on the computational performance of vapor compression cycle simulations. This paper summarizes a Modelica implementation of Spline-Based Table Look-Up Method (SBTL) for fast calculation of refrigerant properties. External C functions are used for faster spline evaluation and inversion. Significant improvement in computation speed was observed without sacrificing accuracy. An SBTL property model of R134a is first validated against a highly accurate Helmholtz energy equation of state (EOS) model. Then the new model was tested rigorously from single function calls, to heat exchanger test bench, to system models of the vapor compression cycle in Modelon's Air Conditioning Library. Finally, an SBTL property model of R1234yf was used in a drive cycle simulation and a shutdown-startup test of two complex air conditioning system models developed at the Ford Motor Company. These system models are running more than twice the speed of the ones using Helmholtz energy EOS.

Keywords: Refrigerant Properties, Equation of State (EOS), Thermodynamic Modeling, Vapor Compression Cycle, Air Conditioning, Spline Interpolation, Computational Performance

1 Introduction

Dynamic simulations of vapor compression cycles often involve significant numbers of function calls to calculate properties of the working fluid. These calculations are typically performed using reference Helmholtz energy (multi-parameter) equation of state (EOS) (Tillner-Roth et al, 1994; Richter et al, 2011) to achieve high accuracy. Short formulation (Span et al, 2003) of Helmholtz energy EOS improves the computational performance, but it does not cover all popular refrigerants, e.g. R1234yf. In Modelon's Air Conditioning Library, both the reference Helmholtz EOS and short Helmholtz EOS are implemented for a wide range of refrigerants.

The two approaches mentioned above have a large impact on the vapor compression cycle simulation speed. First, they have a complicated multi-parameter functional form, which is very costly to evaluate. Moreover, Helmholtz energy EOS uses density and temperature to determine the thermodynamic state, but the system models are usually described by pressure and enthalpy as dynamic states. As a result, internal iteration is needed when the property calculations are performed in a vapor compression cycle simulation.

To address these performance issues, different interpolation methods have been used to approximate refrigerant properties. Extensive literature reviews can be found in the reference (Laughman et al, 2012; Schulze, 2013; Aute et al, 2014) and thus not repeated in this paper. The Spline-Based Table Look-Up Method (Kunick et al, 2015) is chosen to approximate different refrigerant properties in this work because it possesses the following unique features:

- Equidistant grid
- Continuous first derivatives
- Analytic inverse
- Consistent phase boundary definition

The first three features are guaranteed by a specific type of quadratic/biquadratic spline (Späth, 1995). Equidistant grid eliminates the need for searching when evaluating the spline. C^1 continuity is a necessity since some thermodynamic properties are expressed as derivatives, e.g. specific heat capacity, isobaric expansion coefficient, etc. Analytic inverse provides consistent forward and backward calculations without numerical iterations. The last feature avoids chattering around the phase boundary during dynamic simulations. To summarize, the SBTL method takes both function evaluations and system modeling requirements into account, which makes it stand out among different spline interpolation methods for refrigerant property calculation.

Three features in our implementation are tailored for modeling of vapor compression cycle in Modelica: (1) One overall fit over the whole domain is used for 2D splines, instead of fitting several sub-domains. This is to balance data size (hence, loading time) and accuracy. (2) Minimum use of grid transformation. While transforming the grid can improve accuracy, it adds complexity to the implementation and increase the computational cost when taking derivatives and inverting functions. (3) The use of external C functions for the spline evaluation, inversion and derivatives. These calculations are repeated many times in different property functions. Implementing them in C further accelerates the simulations.

The methodology and implementation of the SBTL are further discussed in Section 2. The formulation of the spline interpolation, the data generation process, and the Modelica model structure are covered. The test results of the SBTL property models of R134a and R1234yf are summarized in Section 3. We first compare function calls of the STBL model of R134a to short Helmholtz R134a model in the Air Conditioning Library. The comparisons are then carried out on a heat exchanger testbench and system models. Finally, we tested the SBTL model of R1234yf in a vapor compression cycle model in the Air Conditioning Library before we used it in complex AC system models from Ford described in Section 3.5. The results and performance are compared against the reference Helmholtz R1234yf model (for which short formulation is not available).

2 Methodology and Implementation of Spline-Based Table Look-Up Method (SBTL)

The key concepts of the SBTL method are illustrated in this section using an example of 1D spline. A complete description of the method can be found in the reference (Kunick et al, 2015). The data generation process is also covered here followed by the explanation of the property model structure.

2.1 Overall scheme of the spline functions

The SBTL method uses piece-wise quadratic/bi-quadratic splines to approximate the refrigerant properties. Equidistant gird is used for the spline. So, when evaluating the spline, the interpolating cell is easily known without searching in the whole domain. To enhance the accuracy of the interpolation, transformation can be used on both the independent and dependent variables. In this case, chain rule must be applied properly when calculating derivatives for the transformed variables.

2.1.1 Example of 1D spline and its inverse

The SBTL method distinguishes "node" from "knot". A node is where we have raw data and where the spline intersects with the raw data points. A knot is where two adjacent pieces of splines meet, i.e. the first derivative of the splines are equal. Moreover, a node is the mid-point of two neighboring knots, as shown in Figure 1. For an equidistant series of nodes (or knots) and a given point of evaluation $\bar{x}$, the interval number where it is located is given by

$$i = \text{floor}\left(\frac{\bar{x} - \bar{x}_1{}^K}{\Delta\bar{x}}\right) \qquad (1)$$

where $\bar{x}_1{}^K$ is the first knot of the whole spline and $\Delta\bar{x}$ is the distant between neighboring nodes (or knots). The spline function can then be expressed as:

$$\bar{z}_{\{i\}}(\bar{x}) = \sum_{k=1}^{3} a_{ik}\,(\bar{x} - \bar{x}_i)^{k-1} \qquad (2)$$

where $\bar{x}_i$ is the node in the i^{th} interval and a_{ik} are the spline coefficients in the i^{th} interval. The bar over the independent variable x and dependent variable z indicate that they are transformed variables, which will be discussed in more details in Section 2.1.2. Note that the grid is equidistant in terms of the transformed independent variable $\bar{x}$.

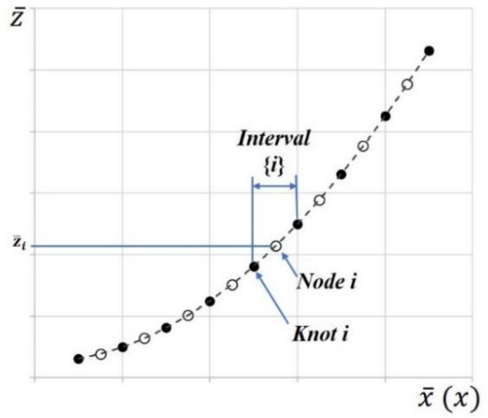


Figure 1. Illustration of the spline interpolation. Knots are represented by solid dots and node by hollow ones.

For a monotonic spline polynomial $\bar{z}_{\{i\}}(\bar{x})$ in the i^{th} interval, the inverse function of the spline is given as:

$$\bar{x}_{\{i\}}{}^{INV}(\bar{z}) = \bar{x}_i + \frac{-a_{i2} \pm \sqrt{a_{i2}{}^2 - 4a_{i3}(a_{i1} - \bar{z})}}{2a_{i3}} \qquad (3)$$

where the sign (±) equals to sign(a_{i2}). An auxiliary spline function $\bar{x}_{\{i\}}{}^{AUX}(\bar{z})$ is also needed to estimate $\bar{x}$, so that we can locate the interval number i.

Spline interpolation in 2D is just an extension of the 1D example. The detailed formulation can be found in the reference (Kunick et al, 2015). The solution algorithm for the spline coefficients are given in the book (Späth, 1995).

2.1.2 Transformations and Derivatives

The equidistant grid enables us to calculate the interval number according to Equation 1, which removes the computational overhead of searching through the whole domain. However, when the function is highly non-linear, as shown in the left plot of Figure 2, we need to increase the number of nodes substantially to better approximate the function by a quadratic spline. This would end up in larger data files, especially for

2D splines (data file size $\sim O(n_{nodes}^2)$), and it would take more time to load the spline coefficient data at initialization.

To balance accuracy and data file size, proper transformations can be applied on both independent and dependent variables. For example, we transformed the pressure coordinate with (base 10) logarithmic function to get better resolution at low pressures.

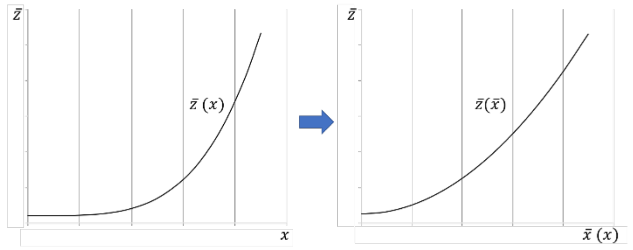


Figure 2. Illustration of coordinate transformation to enhance accuracy with equidistant nodes.

Chain rule must be used for calculation of derivatives. For example, if both the dependent and independent variables are transformed, i.e. $\bar{x} = \bar{x}(x)$ and $\bar{z} = \bar{z}(z, \bar{x})$, then the derivative in the i^{th} interval is given as:

$$\frac{dz_{\{i\}}}{dx} = \frac{d\bar{z}_{\{i\}}}{d\bar{x}} \left(\frac{\partial z}{\partial \bar{z}}\right)_{\bar{x}} \frac{d\bar{x}}{dx} \tag{4}$$

where

$$\frac{d\bar{z}_{\{i\}}}{d\bar{x}} = a_{i2} + 2a_{i3}(\bar{x} - \bar{x}_i) \tag{5}$$

2.2 Data Generation

The property (raw) data is generated using property models in the Air Conditioning Library, more specifically, short Helmholtz model for R134a and reference Helmholtz model for R1234yf. The data is fed to a Python script that solves for the spline coefficients. The coefficient data is then stored as MAT files for later use in the Modelica property model.

The phase boundary of the refrigerant is defined by a 1D spline $T(p_s)$. We have five 2D splines for different properties: temperature $T(p, h)$, density $\rho(p, h)$, entropy $s(p, h)$, dynamic viscosity $\mu(p, h)$, and thermal conductivity $\lambda(p, h)$. For consistency, the bubble and dew enthalpy are expressed as inverse of the 2D temperature spline: $h_{liq}{}^{INV}(p_s, T(p_s))$ and $h_{vap}{}^{INV}(p_s, T(p_s))$. Other properties like specific heat capacity can derived from the 1D and 2D splines (Tummescheit, 2002; Thorade & Saadat, 2013).

We used 100 nodes for the 1D spline and about 120×120 nodes for the 2D splines. For 2D splines, global interpolation is performed for the whole (p, h) domain. To ensure the spline interpolation is accurate in the single phase region up to the phase boundary, the raw data was extrapolated from the single phase region into the 2-phase region. Furthermore, the pressure nodes of the 1D spline $T(p_s)$ overlap with those of the 2D splines (up to the critical point).

2.3 Property Model Structure

The spline coefficient data in the MAT files is loaded once into the memory at initialization of the simulation, so the size of the data file only affects the CPU time at initialization but not during time integration. The structure of the model is shown in Figure 3. The top-level functions for different refrigerant properties are implemented in Modelica. These Modelica functions call the C functions (as external object) that evaluate, invert, and take derivative of the spline.

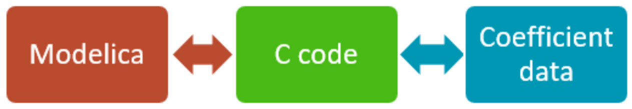


Figure 3. Structure of a thermodynamic property function call in the SBTL model in Modelica.

3 Verification and validation of SBTL for refrigerant property calculations

In this section, comparisons are made between a short formulation of Helmholtz energy EOS (short Helmholtz) model and the SBTL model for R134a. We first look at the CPU time of single function calls and then progress to a heat exchanger test bench and a full system model of vapor compression cycle in the Air Conditioning Library. Finally, we tested an SBTL model of R1234yf in several complex AC system models developed at the Ford Motor Company to evaluate its accuracy and performance in drive cycle simulations. The computer configuration for our tests is shown in Table 1.

Table 1. Configuration of the computer used for testing

Model	Dell Precision M2800 Laptop
Processor	Intel® Core™ i7-4810MQ CPU
RAM	16.0 GB
System	64-bit, x64 based, Windows 10 Pro
Software	Dymola 2018
C compiler	Visual Studio 2012 Express Edition
Solver	Euler (functions), Dassl(systems)
Tolerance	1e-6 (to ensure mass conservation)

3.1 Comparison of function call test results and performance

All the property functions of R134a were tested in the validity range and compared to the short Helmholtz model. Some of the results are listed in Table 2. In the dew enthalpy test, pressure ramped from 0.3 to 39.5

bar. For temperature and density derivative tests, the enthalpy ramped from 150kJ/kg to 500KJ/kg with pressure fixed at 0.3, 0.5, 1, 2, 5, 10, 20 and 39.5 bar. Each test ran for 1s with a fixed step size of 1e-4s, resulting in 1e4 evaluations. CPU time per evaluation was obtained by advanced profiling feature in Dymola.

Table 2. Comparisons of individual function calls

Property	*Relative error in %*	*CPU time short Helmholtz*	*CPU time SBTL*	*CPU Time reduced*
h_{vap}	<0.5%	1.4e-6s	7e-7s	50%
T	<0.03%	>1.2e-5s	<2.0e-6s	>83.3%
$\frac{\partial\rho}{\partial h}\Big\vert_p$	N/A[1]	>2.8e-5s	<3.5e-6s	>87.5%

Each row above represents a certain type of property functions. The dew enthalpy test is representative of saturation properties. The Helmholtz energy EOS uses a cubic spline while the SBTL method inverts the bi-quadratic spline of temperature to obtain saturation enthalpy. The temperature test shows the speed of calculating density, entropy, etc., i.e. evaluation of a 2D spline. The test for partial derivative of density demonstrates the speed-up of calculating partial derivatives, e.g. specific heat capacity, isobaric expansion coefficient, etc. The speed of temperature and the partial derivative functions varies because the computational costs for evaluation in the single-phase region and the two-phase region are different.

The SBTL method is significantly faster with only small deviations in the results. This is partially due to the use of lower order splines. Moreover, the Helmholtz energy EOS uses density and temperature as states, which requires iteration when calling property functions from pressure and enthalpy, while SBTL method simply evaluates spline or its derivatives.

A contour plot of percentage error in density spline evaluation (300×300 points) is shown in Figure 4. White color means the error is below 0.001%. The spline is very accurate in most of the region. Deviation from the reference value locates mainly in the surrounding of the critical point. The maximum error is about 2.4%, which appears inside the two-phase region close to the critical point.

[1]Phase boundary locations are slightly different in the two models, making it hard to compare the results in terms of percentage deviation.

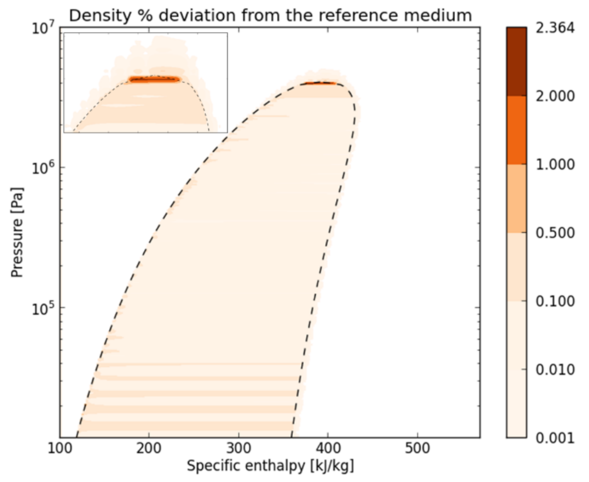


Figure 4. Contour plot of the % deviation of density. Enlarged plot in p = 30-51 bar, h = 330-440 kJ/kg.

3.2 Comparison of heat exchanger test results and performance

System models of vapor compression cycles usually consist of thousands of equations with complicated numerical structures produced by symbolic manipulations. Hence, further proof of concept, beyond property function tests, is required to evaluate the performance of the SBTL model in system level simulations. In a full cycle, the discretized heat exchangers usually have the highest number of property function calls and comprise a large part of the computational cost. Hence, a heat exchanger simulation is a great test case before jumping to a complete vapor compression cycle simulation. An evaporator test bench from the Air Conditioning Library, shown in Figure 5, is used for the test of the SBTL model for R134a. The test bench was simulated for 20s with a ramp in refrigerant mass flow rate increasing from 0.02 to 0.03 kg/s at t = 5s to 7s. Other boundary conditions are kept at constant.

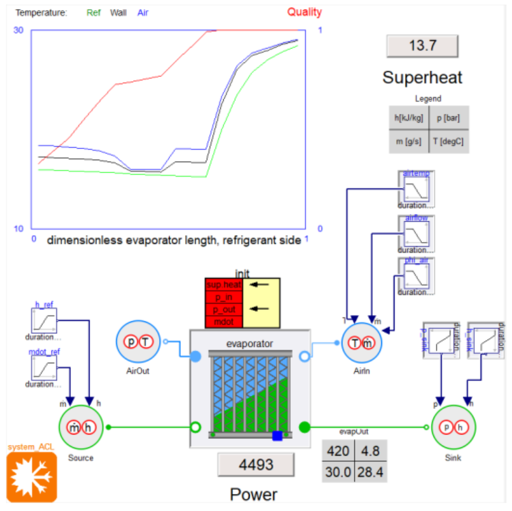


Figure 5. Evaporator test bench in ACL.

Comparisons of the cooling power and air outlet temperature between the SBTL model and the short Helmholtz model can be found in Figure 6. The air outlet temperatures from the two medium models overlap completely and the cooling powers only have small deviations.

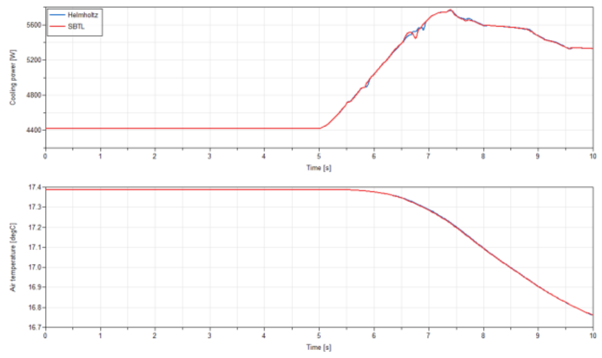


Figure 6. Comparison of cooling power and air outlet temperature. Blue – short Helmholtz, Red – SBTL.

Figure 7 shows the CPU time comparisons. The red curves are CPU time after initialization while the blue curves include CPU time of initialization of the model. Dotted curves are for short Helmholtz model while solid ones are for SBTL model. As we can see, the SBTL model was running twice the speed both at initialization and during the transient run.

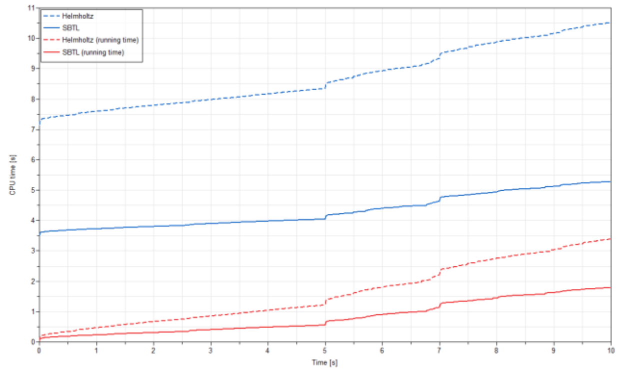


Figure 7. Comparison of CPU time. Dotted – short Helmholtz, Solid – SBTL; Red – CPU time after initialization, Blue – Total CPU time.

The heat exchanger test results demonstrate the speed-up provided the SBTL model beyond single function calls, and they serve as good indicators of the performance improvement in a full vapor compression cycle, as discuss in the later sections.

3.3 Comparison of system models in the Air Conditioning Library results and performance

The pull-down test from the Air Conditioning Library, depicted in Figure 8, is used to evaluate the performance of the SBTL model of R134a. The model is run for 4000s to get to a steady state. Results and CPU time are benchmarked against the short Helmholtz R134a model.

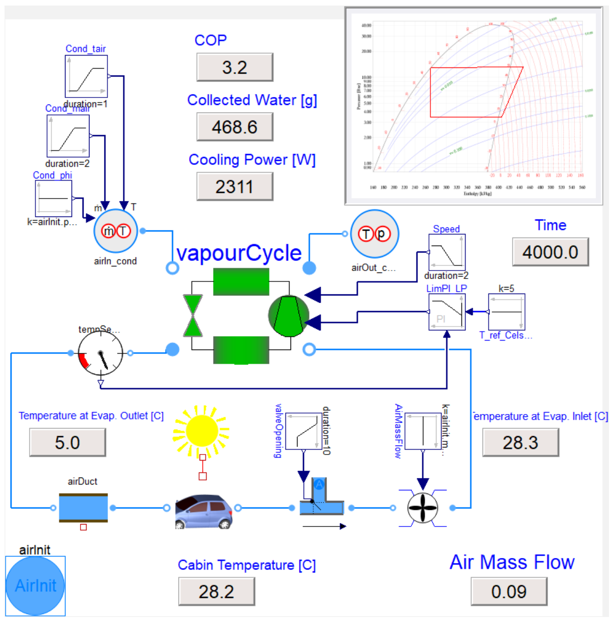


Figure 8. Pull-down test from ACL for an air conditioning cycle connection with vehicle cabin.

Deviations of some key results (trajectory during the whole simulation) from the short Helmholtz R134a model are listed in Table 3. The SBTL model replicated the result of the benchmark model very accurately.

Table 3. Deviations of key results in the pull-down test

Key results	*Deviation in %*
Cooling Power	< 0.4
Refrigerant mass flow rate	< 0.2
Cabin temperature	< 0.002

The CPU time plots are shown in Figure 9. The upper one is for the entire 4000s simulation and the lower one zooms into the first 100s when most of the dynamics happened. The SBTL model reduced the CPU time by about 33%.

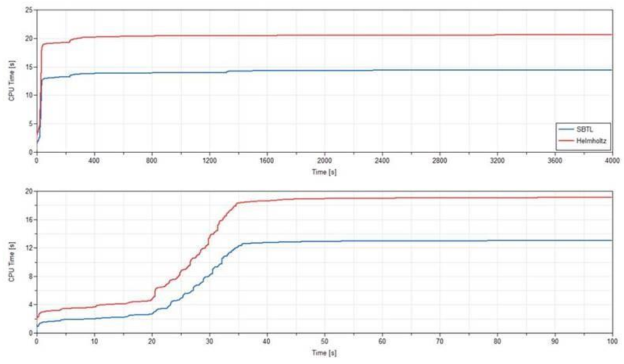


Figure 9. CPU time comparison of the pull-down test. Upper - entire simulations, Lower - first 100s.

All the tests discussed so far compare the SBTL model to the short Helmholtz model which is faster than reference state Helmholtz EOS. However, not every refrigerant in Air Conditioning Library has a short formulation. For example, R1234yf, proposed as a replacement for R1234a in automotive air conditioning systems, only has reference state Helmholtz EOS. Hence, we expected a larger performance improvement when using the SBTL method for R1234yf systems. The orifice cycle model from the Air Conditioning Library (Figure 10) was simulated for 180s to further study the performance improvement by SBTL model for R1234yf.

Table 4. Deviations of key results in the R1234yf orifice cycle simulations

Key results	*Deviation in %*
Cooling Power	< 0.1
Refrigerant mass flow rate	< 0.02
Cabin temperature	< 0.001

The accuracy of the model is verified by comparing the dynamic trajectory of some key results, as listed in Table 4.As shown in the CPU time plot in Figure 12, the orifice cycle model with SBTL R1234yf ran twice as fast as the reference.

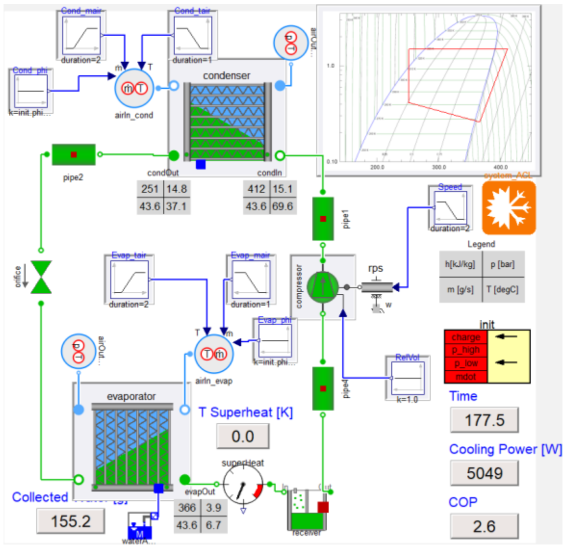


Figure 10. Orifice cycle model using R1234yf in ACL.

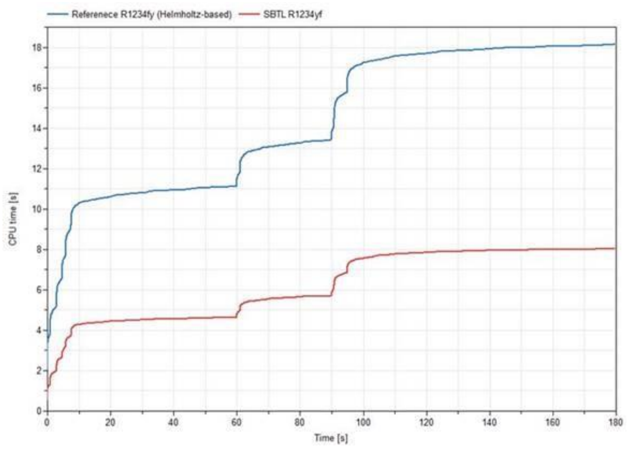
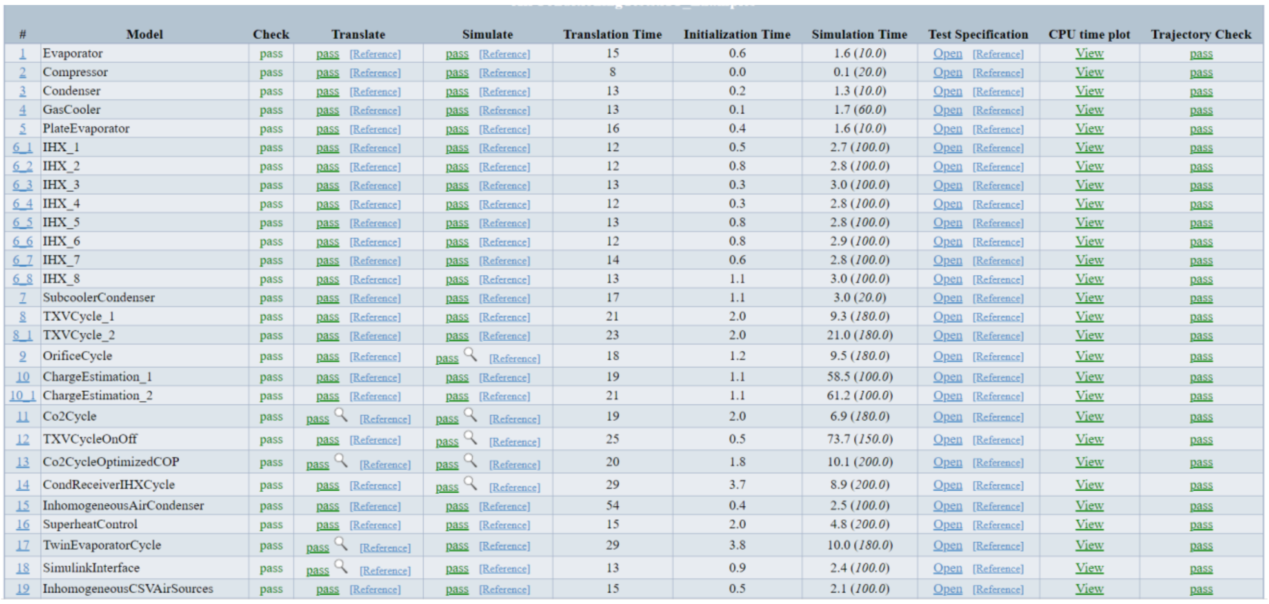


Figure 12. CPU time comparison of the orifice cycle simulations. Blue - Reference Helmholtz, Red – SBTL.



#	Model	Check	Translate	Simulate	Translation Time	Initialization Time	Simulation Time	Test Specification	CPU time plot	Trajectory Check
1	Evaporator	pass	pass [Reference]	pass [Reference]	15	0.6	1.6 (*10.0*)	Open [Reference]	View	pass
2	Compressor	pass	pass [Reference]	pass [Reference]	8	0.0	0.1 (*20.0*)	Open [Reference]	View	pass
3	Condenser	pass	pass [Reference]	pass [Reference]	13	0.2	1.3 (*10.0*)	Open [Reference]	View	pass
4	GasCooler	pass	pass [Reference]	pass [Reference]	13	0.1	1.7 (*60.0*)	Open [Reference]	View	pass
5	PlateEvaporator	pass	pass [Reference]	pass [Reference]	16	0.4	1.6 (*10.0*)	Open [Reference]	View	pass
6_1	IHX_1	pass	pass [Reference]	pass [Reference]	12	0.5	2.7 (*100.0*)	Open [Reference]	View	pass
6_2	IHX_2	pass	pass [Reference]	pass [Reference]	12	0.8	2.8 (*100.0*)	Open [Reference]	View	pass
6_3	IHX_3	pass	pass [Reference]	pass [Reference]	13	0.3	3.0 (*100.0*)	Open [Reference]	View	pass
6_4	IHX_4	pass	pass [Reference]	pass [Reference]	12	0.3	2.8 (*100.0*)	Open [Reference]	View	pass
6_5	IHX_5	pass	pass [Reference]	pass [Reference]	13	0.8	2.8 (*100.0*)	Open [Reference]	View	pass
6_6	IHX_6	pass	pass [Reference]	pass [Reference]	12	0.8	2.9 (*100.0*)	Open [Reference]	View	pass
6_7	IHX_7	pass	pass [Reference]	pass [Reference]	14	0.6	2.8 (*100.0*)	Open [Reference]	View	pass
6_8	IHX_8	pass	pass [Reference]	pass [Reference]	13	1.1	3.0 (*100.0*)	Open [Reference]	View	pass
7	SubcoolerCondenser	pass	pass [Reference]	pass [Reference]	17	1.1	3.0 (*20.0*)	Open [Reference]	View	pass
8	TXVCycle_1	pass	pass [Reference]	pass [Reference]	21	2.0	9.3 (*180.0*)	Open [Reference]	View	pass
8_1	TXVCycle_2	pass	pass [Reference]	pass [Reference]	23	2.0	21.0 (*180.0*)	Open [Reference]	View	pass
9	OrificeCycle	pass	pass [Reference]	pass [Reference]	18	1.2	9.5 (*180.0*)	Open [Reference]	View	pass
10	ChargeEstimation_1	pass	pass [Reference]	pass [Reference]	19	1.1	58.5 (*100.0*)	Open [Reference]	View	pass
10_1	ChargeEstimation_2	pass	pass [Reference]	pass [Reference]	21	1.1	61.2 (*100.0*)	Open [Reference]	View	pass
11	Co2Cycle	pass	pass [Reference]	pass [Reference]	19	2.0	6.9 (*180.0*)	Open [Reference]	View	pass
12	TXVCycleOnOff	pass	pass [Reference]	pass [Reference]	25	0.5	73.7 (*150.0*)	Open [Reference]	View	pass
13	Co2CycleOptimizedCOP	pass	pass [Reference]	pass [Reference]	20	1.8	10.1 (*200.0*)	Open [Reference]	View	pass
14	CondReceiverIHXCycle	pass	pass [Reference]	pass [Reference]	29	3.7	8.9 (*200.0*)	Open [Reference]	View	pass
15	InhomogeneousAirCondenser	pass	pass [Reference]	pass [Reference]	54	0.4	2.5 (*100.0*)	Open [Reference]	View	pass
16	SuperheatControl	pass	pass [Reference]	pass [Reference]	15	2.0	4.8 (*200.0*)	Open [Reference]	View	pass
17	TwinEvaporatorCycle	pass	pass [Reference]	pass [Reference]	29	3.8	10.0 (*180.0*)	Open [Reference]	View	pass
18	SimulinkInterface	pass	pass [Reference]	pass [Reference]	13	0.9	2.4 (*100.0*)	Open [Reference]	View	pass
19	InhomogeneousCSVAirSources	pass	pass [Reference]	pass [Reference]	15	0.5	2.1 (*100.0*)	Open [Reference]	View	pass

Figure 11. A part of the ACL regression test report. SBTL property model was used in the tests and the results were compared against reference results obtained by the Helmholtz property models.

3.4 Some comparisons from our full suite of the Air Conditioning Library regression tests

To ensure the robustness and accuracy of the SBTL property models, we used them in our existing regression tests of the Air Conditioning Library and compared to the reference results generated by the Helmholtz property model. A test was considered "pass" only if it compiled, simulated and produced results within a tight tolerance of the references. A small portion of the full regression test suite is shown in **Figure 11**.

3.5 Comparison of Ford AC system models results and performance

In this section, simulations are performed on two complex AC system models developed at Ford Motor Company R1234yf is used in both systems. Figure 13 depicts an AC system with two evaporators and one chiller connected in parallel in the loop. Simulations of the SC03 drive cycle were performed on this model. The cooling power of the evaporators and the chiller can be found in Figure 14. The SBTL model replicates the results from the Helmholtz model very closely. Figure 15 shows the comparison of CPU time. The SBTL model took only 509s to run, which is 85% of the real-time (598s), and it saves more than 60% of CPU time compared to the Helmholtz model.

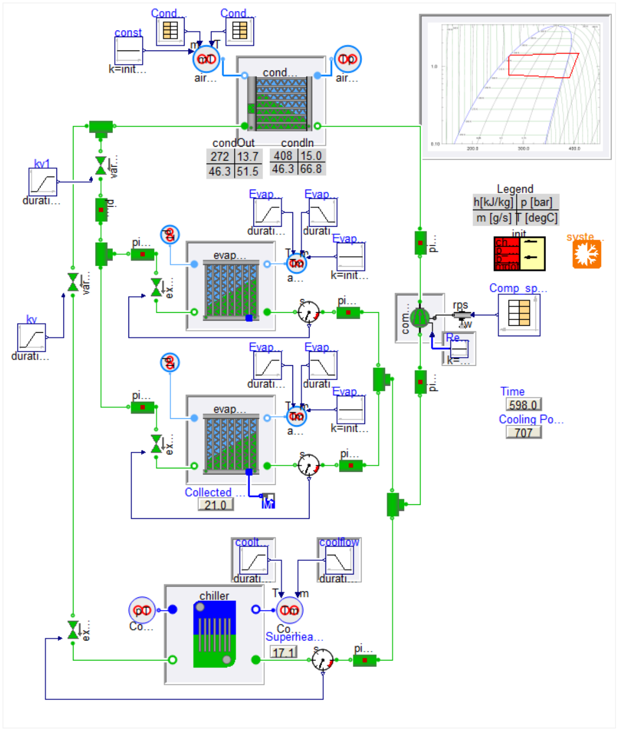


Figure 13. AC system with two evaporators and one chiller connected in parallel in the loop.

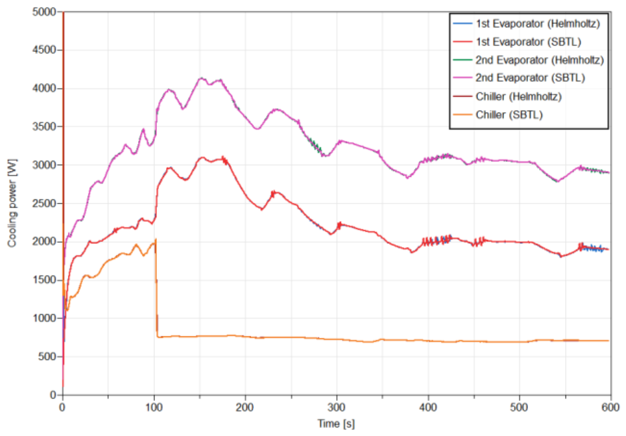


Figure 14. Cooling power of the evaporators and the chiller using Helmholtz property model and SBTL model.

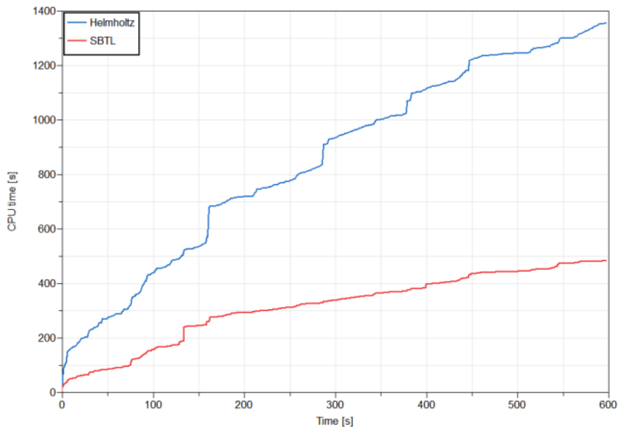


Figure 15. CPU time comparison of the AC system model shown in Figure 13.

Figure 16 is a vapor compression cycle with a chiller connected to a battery cooling loop. In the simulation, the compressor was off at t = 0s, and the refrigerant loop was initialized with a certain mass flow rate, i.e. the simulation started at the moment when the compressor was turned off. The compressor was turned on again when the battery temperature (cooled by the cooling loop) is above a certain threshold. This shut down and startup test is challenging because of low refrigerant flowrate when the compressor is off and the fast dynamics when it restarts.

The compressor speed and refrigerant mass flow rate are plotted in Figure 17. The SBTL model predicts the mass flow well even during the fast transients after the compressor restarted. CPU time comparison can be found in Figure 18. The SBTL model saves more than 70% of the CPU time.

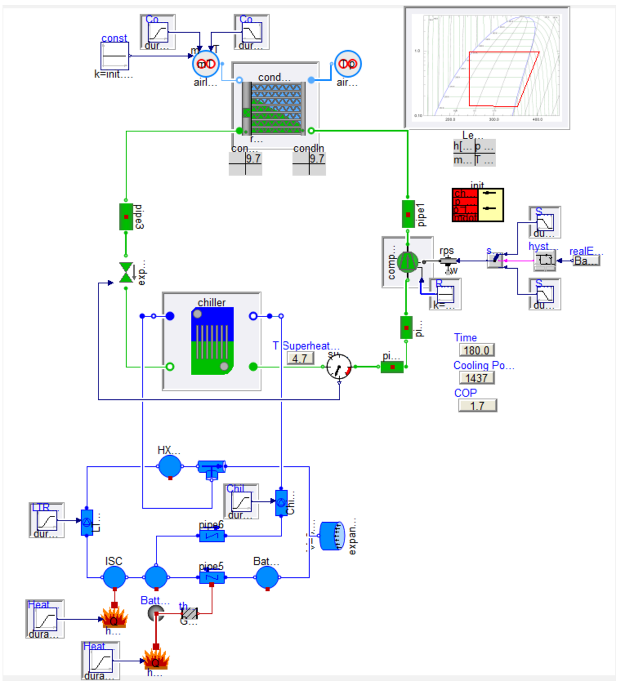


Figure 16. R1234yf vapor compression cycle with a chiller connected to a battery cooling loop.

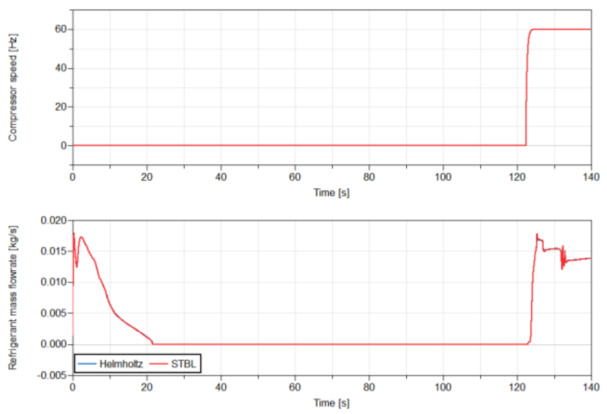


Figure 17. Compressor speed and refrigerant mass flow rate.

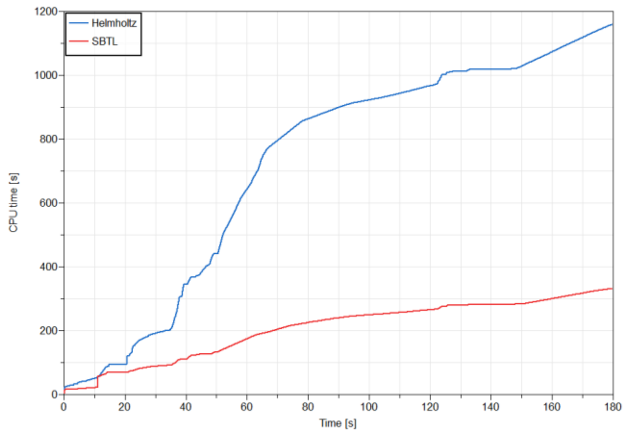


Figure 18. CPU time comparison for the model shown in Figure 16.

4 Conclusions

This paper summarizes an implementation of the SBTL method in Air Conditioning Library for fast calculation of refrigerant properties using Modelica language. The SBTL method, the data generation process, and the property model structure are explained. The SBTL refrigerant property models demonstrate significant improvement in computational speed in single function calls. In system simulations of AC cycle with R134a, the SBTL model cut the CPU time by 33% compared to the short Helmholtz model. The complex AC system models from Ford Motor Company run twice the speed with SBTL model of R1234yf than with reference Helmholtz model. The SBTL models for R134a and R1234yf will be available in the upcoming 2018.2 release (version 1.17) of Modelon's Air Conditioning Library.

References

Tillner-Roth, R. and Baehr, H.D. An international standard formulation of the thermodynamic properties of 1,1,1,2-tetrafluoroethane (HFC-134a) for temperatures from 170 K to 455 K at pressures up to 70 Mpa. *Journal of Physical and Chemical Reference Data*, 23(5), 657-729, 1994.

Span, R. and Wagner, W. Equations of state for technical applications. I. Simultaneously optimized functional forms for nonpolar and polar fluids. *International journal of thermophysics*, 24(1), 1-39, 2003.

Richter, M., McLinden, M.O., and Lemmon, E. W. Thermodynamic Properties of 2, 3, 3, 3-Tetrafluoroprop-1-ene (R1234yf): Vapor Pressure and p–ρ–T Measurements and an Equation of State. *Journal of Chemical & Engineering Data*, 56(7), 3254-3264, 2011.

Laughman, C., Zhao, Y., and Nikovski, D. Fast Refrigerant Property Calculations Using Interpolation-Based Methods. *International Refrigeration and Air Conditioning Conference*. Paper 1344, 2012.

Schulze, C. W. A contribution to numerically efficient modelling of thermodynamic systems. *PhD Theses*, 2013.

Aute, V. and Radermacher, R. Standardized Polynomials for Fast Evaluation of Refrigerant Thermophysical Properties. *International Refrigeration and Air Conditioning Conference*. Paper 1499, 2014.

Kunick, M., and H. J. Kretzschmar. Guideline on the fast calculation of steam and water properties with the spline-based table look-up method (SBTL). *Technical report, The International Association for the Properties of Water and Steam*, Moscow, Russia, 2015.

Späth, H. One dimensional spline interpolation algorithms. AK Peters/CRC Press, 1995.

Späth, H. *Two dimensional spline interpolation algorithms*. AK Peters, Ltd., 1995.

Tummescheit, H. Design and implementation of object-oriented model libraries using Modelica. *PhD Theses*, 2002

Thorade, M. and Saadat, A Partial derivatives of thermodynamic state properties for dynamic simulation. *Environmental earth sciences*, 70(8), pp.3497-3503, 2013.