A Fast Tabulated Equation of State for CFD Simulation of Two-phase Flows

Antoine Metsue\textsuperscript{1,2,*}, Yann Bartosiewicz\textsuperscript{2}, Sébastien Poncet\textsuperscript{1}
\textsuperscript{1Mechanical Engineering Department, Université de Sherbrooke, Sherbrooke, Canada}
\textsuperscript{2Institute of Mechanics, Materials, and Civil Engineering, Université catholique de Louvain, Louvain-la-Neuve, Belgium}
\textsuperscript{*antoine.metsue@usherbrooke.ca}

Abstract—One of the challenging aspects when performing CFD of multi-phase flows is the modeling of the fluid equation of state, used to close the system of equations. Firstly, one must define a model to handle the two-phase behavior of the fluid, i.e., under the saturation curve, as it is generally not described by the equation of state itself, that is only valid outside of the saturation dome. Secondly, the related numerical method during the simulation itself should be quite fast as it will be called as many times as there are point/cells in the mesh at each iteration. In this work, a novel tabulated equation of state method is presented, with the intent of performing fast and accurate CFD of flashing nozzles/ejector-type devices. The table is built using the equation of state provided by the REFPROP library and is here applied to two-phase CO\textsubscript{2}, although the method could be used for any fluid. Likewise, any thermodynamic relation that is implemented in REFPROP can be generated as a tabulated method. A bicubic interpolation scheme is used with local refinement of the meshing in the thermodynamic domain. For the purpose of this work, the tabulated method was coupled to the SU2 density-based solver. Results indicate that the present method is about 10 times faster than previously tabulated methods for CO\textsubscript{2}, and about 500 times faster than the Span-Wagner equation of state, which is the reference for that fluid.

Keywords—Tabulated method, equation of state, CO\textsubscript{2}, two-phase, REFPROP, COOLPROP, SU2

I. INTRODUCTION

In order to close the system of equations in a CFD simulation, one requires an equation of state (EoS). In general, this EoS is used to compute fluid properties at vapor, liquid, supercritical, metastable and two-phase equilibrium states. As demonstrated by Raman and Kim [1], the computation of those properties can strongly influence the results of CFD simulations. Note that the computation time for EoS should be reasonable. As stated by De Lorenzo et al. [2], there are three ways to achieve this goal: to use a simple EoS, to employ an iterative algorithm adopting an accurate EoS or to use tabulated methods.

Smolka et al. [3] presented a model of the compressible transonic single- and two-phase flow of a real fluid within an ejector. The authors validated their model using the perfect gas (PG) EoS, both for the single-phase refrigerants and for the two-phase CO\textsubscript{2} even though the PG EoS is not suited for predicting such complex mixtures [4]. The stiffened gas (SG) EoS is an alternative EoS defined by the Helmholtz free energy. It has the advantage of being relatively simple and accurate and is based on the linearization of a more general EoS [5]. Although simulations of CO\textsubscript{2} flows have been performed using the SG EoS [5, 6], the latter remains limited because its accuracy range is limited to the vicinity of the reference linearization point. Another alternative is the Peng-Robinson (PR) EoS, that has been used by Brown et al. [7] to simulate high pressure CO\textsubscript{2} pipelines. Nevertheless, it is widely accepted that the currently most accurate EoS for CO\textsubscript{2} is the Span-Wagner (SW) EoS [8], that is based on hundreds of parameters that are fitted by extensive experimental data. This EoS has been implemented into the REFPROP [9] and COOLPROP [10] thermophysical property libraries. Despite its high accuracy, the SW EoS requires high CPU times, which is the main reason why so called tabulated EoSs are being preferred. With those methods, the time-consuming iterative procedure for the EoS is usually performed at the beginning of the computation to create a table that will be used in the later stages of the simulation, by interpolating the tabulated values.

A CO\textsubscript{2} lookup table method based on the Perturbed-Chain Statistical Associating Fluid Theory EoS has been proposed by Brown et al. [11]. Giacomelli et al. [12] constructed lookup tables in the \((v - e)\) (specific volume - specific internal energy) space from REFPROP using an in-house Matlab code, to perform ejector simulations using ANSYS Fluent. The authors implemented a bi-linear interpolation scheme to retrieve the adequate values of density and speed of sound at each cell of the domain. Ameli et al. [13] later implemented a similar lookup table in FORTRAN and coupled it to the ANSYS CFX 17.0 solver to investigate the condensation process in transcritical CO\textsubscript{2} nozzles. Fang et al. [4] developed a similar FORTRAN lookup table for CO\textsubscript{2} by adapting the work of De Lorenzo et al. [2] for water. The tabulated method uses bilinear interpolation and has a speed-up factor of about 66 compared to the original SW EoS. Furthermore, it has been coupled, by the same authors [14], to a density-based OpenFOAM solver to simulate transcritical CO\textsubscript{2} ejector flows. Later, Föll et al. [15] developed

ID 357 – CFD symposium 1
a tabulated method based on piecewise polynomials with adaptative refinement in the state space and coupled it to the FORTRAN CFD solver FLEXI [17]. The authors then simulated a freestream of nitrogen in a periodic box. Although their generation algorithm is costly (570 s over 96 processors), they estimated a time gain of about 230 times compared to a direct evaluation of the EoS in COOLPROP. Lastly, Rubino et al. [16] developed a look-up table based on unstructured grids with mesh refinement. Their method based on a bilinear interpolation scheme was coupled to the non-ideal CFD SU2 solver. They found out that an unstructured grid was beneficial in terms of computation time for coarse meshed only.

In this work, a new general tabulation algorithm is presented. The tabulated method combines for the first time a piecewise bicubic interpolation of the thermodynamic properties with local refinement of the interpolation grid based on a required precision. Indeed, because look-up tables can quickly require much memory [15], a good compromise between precision and storage size was found by using the bicubic interpolation. This method, based on the equation of state provided by the REFPROP/COOLPROP libraries, can be applied to any type of fluid and the relations can be built from any state space given its implementation in said thermo-libraries. Although the homogeneous equilibrium hypothesis was made in the present context, the method is not restricted to the latter and can be used to generate table for metastable properties. Since the target of this library is to be used with an implicit solver, the required thermodynamic derivatives have also been implemented. The look-up table features local mesh refinement to ensure a given absolute precision in the whole state space. With the aim of providing the best versatility of the table, the latter is generated once and then stored on dedicated files, which are then read at the beginning of the actual CFD simulation. This means that this table can be used in any programing language without modification since only the interpolation function, which is simple and quite short, needs to be implemented in that said language.

II. METHODS

A. Equation of state

As mentioned in the previous section, the equation of state that is used in the present work is provided by the REFPROP/COOLPROP libraries. As the selected solver for the present study is the SU2 solver, one will focus in what follows on the thermodynamic relations that are required by this solver. The latter being suited for non-ideal CFD (i.e., real gas CFD), the state is contained within a Fluid object and the equation of state is implemented in the related Fluid class. The thermodynamic relations required by the solver must then be implemented. In this case, since the solver is density-based, the relation needed for closure is of the type:

\[ z = f(\rho, e), \] (1)

where \( z \) is any thermodynamic property, \( \rho \) is the density and \( e \) is the specific internal energy. Depending on the desired features of the solver, thermodynamic relations are also required in state spaces different than \((\rho - e)\), e.g., in \((p - T)\) and \((h - s)\) state spaces for the computation of total boundary conditions, where \( p \) is the pressure, \( T \) is the temperature, \( h \) is the specific enthalpy and \( s \) is the specific entropy.

The code that is implemented to generate the table is written as a Python routine and the properties are retrieved through the Python wrapper for COOLPROP, that offers a REFPROP interface as well. By defaults, the properties are retrieved from REFPROP except in the case of the thermodynamic derivatives, that are only available in COOLPROP. These derivatives are however limited to the single-phase region and two-phase properties must be reconstructed. In this work, two-phase properties are calculated with the homogeneous equilibrium hypothesis, that is:

\[ p_l = p_v = p_{sat}, \]
\[ T_l = T_v = T_{sat}, \]
\[ u_l = u_v = u_{sat}, \]
\[ g = g_v = g_{sat}, \] (2)

where \( u \) is the velocity, \( g \) is the Gibbs free enthalpy and the \( l, v \) and \( sat \) subscripts refer to the liquid, vapor and saturated states, respectively. The quality \( x \) is then defined as

\[ x = \frac{v - v_l}{v_v - v_l} \] (3)

where \( v \) is the specific volume. The enthalpy, entropy and energy \((z)\) are then reconstructed under the saturation dome as:

\[ z = (1 - x) z_l + x z_v. \] (4)

Derivative-based properties can then be reconstructed as well from the saturation properties and the derivatives along the saturation line.

B. Interpolation method

![Figure 1. Schematic of a cell element](image-url)

Figure 1. Schematic of a cell element
A bicubic interpolation scheme is implemented in the state space. This feature allows good precision whilst requiring limited memory storage compared to more complex approximation methods or even bilinear interpolation as a bicubic function reduces the amount of required cell elements by two orders of magnitude while requiring only 4 times more memory per cell, compared to the bilinear interpolation. Indeed, the latter requires either 4 polynomial coefficients/evaluations of the function, and the bicubic interpolation requires 16.

The state space is discretized by a number of rectangular cells until a given absolute interpolation precision is reached for the required thermodynamic variable. The mesh generation algorithm in the state space is described in the following subsection. Fig. 1 shows a schematic of a cell element. Each cell is defined by its boundaries in (x, y) and the values at the 16 equally spaced nodes (black dots in the figure) are evaluated to calculate the bicubic polynomial F:

\[ F = X \ A \ Y, \]  

where \( X \) is the 4x4 matrix where each line is of the form \( [1, x_i, x_i^2, x_i^3] \), with the \( i \)th subscript ranging from 1 to 4, which corresponds to the abscissa of the 16 passage points (Fig. 1). Matrix \( Y \) is reciprocally the same for the ordinate but transposed, and \( F_{ij} = f(x_i, y_j) \) is the evaluation of the thermodynamic function. The 16 polynomial coefficients \( A_{ij} \) can then be retrieved by inversion of the matrices:

\[ A = X^{-1} \ F \ Y^{-1}. \]  

To speed up the generation process and avoid a loss in accuracy (due to finite machine precision) for large values of \((x,y)\), the cell boundaries are normed, thereby, \( X^{-1} = Y^{-1} = I \) is constant.

During the CFD simulation, the 16 coefficients of \( A \) are retrieved from the look-up table and the thermodynamic property is computed. Note that the coordinates in the state space must also be normed by the cell dimensions, i.e.,

\[ x' = (x - x_1)/(x_2 - x_1), \quad y' = (y - y_1)/(y_2 - y_1), \]  

with \( x' \) and \( y' \) the normed coordinates.

C. Mesh generation algorithm

1) Lookup table boundaries

The present lookup table is generated with the aim of performing CFD of transcritical CO\(_2\) nozzle and ejector devices. Therefore, the state space boundaries have been chosen to englobe the range in which those devices should operate. However, it is necessary to provide some margins because the numerical solvers can oscillate during the solution convergence process. Fig. 2 shows the \( p-h \) diagram of CO\(_2\) that constitutes the validity boundary for the lookup table that is considered here. The green area illustrates a typical transcritical CO\(_2\) cycle and is indicative of the range in which the table will be used.

In this particular case, the table boundaries were chosen in the \( p-h \) diagram because it offers good readability for application in refrigeration thermodynamic cycles and is easy to interpret compared to less usual diagram such as the \( \rho - e \) diagram. The bottom/top boundaries were chosen as 0.52/50 [Mpa]. The low pressure corresponds to the triple point pressure.

The left boundary is the enthalpy at the triple point whereas the right boundary corresponds to the enthalpy at 800 K for the low pressure. Those boundaries apply for all tables generated in this particular case (i.e., transcritical CO\(_2\) applications) and for all of the thermodynamic relations whether the state space is \( \rho - e \) or another one. The method is however not restricted to such kind of boundaries and the user is free to input other ones.

![P-h diagram of CO2](image)

**Figure 2**. P-h diagram of CO2. The table is generated for the whole domain. The iso-quality lines are traced in red.

2) Recursive algorithm

The mesh is generated using object-oriented programming in Python, where each cell (Section 2.B) is an instance of a Cell object, that contains the cell coordinates, polynomial interpolation coefficients and cell number as attributes. Fig. 3 shows a flowchart of the corresponding algorithm.

The routine consists in creating an instance of a cell object that will constitute the grid. The constructor is called with specified cell boundaries, which correspond to the state-space domain, an absolute error tolerance for the tabulated thermodynamic property and a first cell number, by default set to 0. The constructor then calculates matrix \( A \) with Eq. 6 and if the cell is within the state space domain or overlapping it, an interpolation error is approximated. This error is computed by taking the maximum absolute error over 9 verification points that are located in between the 16 passage points (see the red dots in Fig. 1). If the error is higher than the specified tolerance, the cell is subdivided into 4 sub-cells. The constructor is then called recursively, and the 4 sub-cells (green dash lines in Fig. 1) are instantiated as attributes of the current cell; for each of those 4 cells, the constructor is called with a halved range, the same tolerance and a new unique cell number. This process goes until the tolerance criterion is met, then the cell data is stored in an external text file. Each line of the text file corresponds to a cell and is composed of the cell number, its \( x \) and \( y \) boundaries in the state space, the 16 polynomial coefficients and the 4 sub-cell numbers. If the cell is at the bottom of the tree, there are no
sub-cells and -1 is stored instead, which acts as a flag for the later reading process.

3) Table issues management

Because the thermodynamic relations provided by COOLPROP/REFPROP are not always able to provide a solution, e.g., near the critical point, the algorithm should be able to handle such cases without storing non-physical values. The strategy adopted in the present work is twofold: (i) if no value can be computed at any of the 16 interpolation points within the cell, this current cell is deleted and the lookup table falls back on the upper cell, though the precision criterion may not be met; (ii) if some values can be computed but not for all of the 16 interpolation points, the cell is further subdivided unless no values can be computed in one of its 4 cells (i.e., case (i)).

![Figure 3. Flowchart of the grid generation algorithm.](image)

In addition, because there can be some quite high gradients near the saturation line, one can also choose to relax the error tolerance if the cell dimensions are getting below a certain threshold, in order to avoid building a prohibitively large lookup table.

Even though those strategies relax the error tolerance, the error is located in a very small range of the state space, often located in the near vicinity of the critical point. The actual error of the generated table is thus checked a posteriori to ensure that it stays reasonable in that said region. In addition, note that during the CFD simulation itself (here for CO₂ ejector simulations), some thermodynamic states may pass through the region of less accuracy, but usually only during the convergence process.

III. RESULTS

A. Grid quality and error verification

![Figure 4. Pressure field in [bar] in color and related grid in the \( \rho - e \) state space.](image)

Figures 4 and 5 display the grids generated for the pressure and temperature in the \( \rho - e \) state space, respectively. The mesh in Fig. 4 was generated by imposing a maximum interpolation error of 0.1 [Pa] with relaxation to 0.1 [kPa] if the cell width is below 0.5 \([\text{kg/m}^3]\). The grid is composed of 175,637 cells/sub-cells elements and requires roughly 35 MB of RAM during the CFD simulation. In Fig. 5, the tolerance is set to 0.0001 [K] with relaxation to 0.001 [K] with the same cell width threshold. Likewise, the grid is composed of 96,589 cells/sub-cells elements and requires roughly 20 MB of RAM during the CFD simulation.

Although for both grids most of the cells are located along the saturation curve, the pressure grid requires more elements in the region of low energy and high density. Indeed, in this region the fluid is in the liquid phase and the pressure varies quite strongly with both state variables.

Figures 6 and 7 show the relative error defined as:

\[
\epsilon = \left| \frac{z_{\text{interp}} - z_{\text{sol}}}{z_{\text{sol}}} \right| ,
\]  

(8)
for pressure and temperature, respectively. The plots were generated by taking $10^6$ linearly spaced points in the $\rho - e$ state space. Only the validity range of the lookup table is shown. The relative error is mostly within the range $10^{-6} - 10^{10}$ for the pressure. Actually, the higher errors, of the order of $10^{-4}$ for the pressure and $10^{-6}$ for the temperature are located in the very near vicinity of the saturation line.

**Figure 5.** Temperature field in [K] in color and related grid in the $\rho - e$ state space.

**Figure 6.** Relative error $\epsilon$ of the pressure grid (Fig. 4).

**Figure 7.** Relative error $\epsilon$ of the temperature grid (Fig. 5).

**B. Test case: transcritical CO$_2$ ejector**

As mentioned previously, the present lookup-table was coupled to the SU2 [18] solver. The latter was used to perform a wall-resolved Reynolds-Averaged Navier-Stokes (RANS) simulation of the transcritical CO$_2$ ejector of Elbel et al. (private communication). The convective fluxes are solved with the flux-splitting Roe scheme and second-order accuracy is achieved through the MUSCL approach. The spatial gradients are computed by the Green-Gauss method. The Euler implicit time integration scheme is used with the $k-\omega$ SST turbulence closure scheme. The 2D axisymmetric mesh is composed of about 200,000 rectangular elements. The grid is mainly structured except in the near vicinity of the primary nozzle exit, where it is highly refined to correctly capture the supersonic structures that can be observed in the Mach number field shown in Fig. 8. For this particular simulation, the primary inlet conditions are fixed to $p_p = 9.066$ [MPa], $T_p = 310.95$ [K], and the secondary inlet conditions are fixed to $p_s = 4.647$ [MPa], $T_s = 291.05$ [K]. The outlet pressure is set to $p_{out} = 4.990$ [MPa].

The computation time was compared when using the lookup table method of Fang et al. [4] and using the present lookup table. Both simulations ran with similar solver configuration and were set to perform 100,000 (pseudo-)time steps on 80 processors. A significant computation time improvement was observed, as the simulation performed with the table of Fang et al. [4] took 109 120 seconds to simulate, while the simulation with the present table took only 9 334 seconds, thereby yielding an approximate speedup of about 12 times.

**Figure 8.** Mach number field within the ejector. The geometry has been stretched twofold in the vertical direction.

ID 357 – CFD symposium
IV. Conclusion

A new method for generating fast tabulated thermodynamic equations of state was presented. The tabulated method can easily be used in any language and the tables themselves have to be generated only once. Tables are built from the thermodynamic relations provided by the COOLPROP and REFPROP libraries; any thermodynamic relation provided by said libraries can be built as a table. The method generates grids with local refinement from a given error tolerance between the bicubic interpolation and the exact solution.

The homogeneous equilibrium hypothesis was used to generate lookup tables for transcritical CO₂. The tables accuracy was verified, and the method was subsequently coupled to the real-gas CFD SU2 solver. The latter was used to perform a simulation of a flashing ejector and the computation time was compared to using the bilinear lookup-table method of Fang et al. [4]. Results showed a significant speedup, the present tabulated method being roughly 12 times faster.

Future works are foreseen, such as using the method to generate tables of metastable properties, that would allow to use more advanced two-phase models, such as the Homogeneous Relaxation Model. The method could also be improved by building the extrapolation of the relevant thermodynamic property outside of the validity region in order to allow easy convergence of the CFD solver. Lastly, thanks to the speedup provided by the tabulated method, large computation meshes such as full 3D wall-resolved simulations of ejector flows are now accessible, as well as parametric studies for design and optimization of the geometry.

Acknowledgment

The NSERC chair on industrial energy efficiency established in 2019 with the support of Hydro-Québec (laboratoire des technologies de l’énergie), Natural Resources Canada (CanmetEnergy-Varennes) and Emerson Commercial & Residential Solutions is here gratefully acknowledged. The calculations have been performed using the HPC facilities of the Digital Research Alliance of Canada.

References


