Development of Extrinsic Functions for Optimal Synthesis and Design—Application to Distillation-based Separation Processes

Juan I. Manassaldi[1], Miguel C. Mussati[1,2], Nicolás J. Scenna[1], Sergio F. Mussati[1,2,*]

[1] CAIMI Centro de Aplicaciones Informáticas y Modelado en Ingeniería, Universidad Tecnológica Nacional, Facultad Regional Rosario, Zeballos 1346, S2000BQA Rosario, Argentina.

[2] INGAR Instituto de Desarrollo y Diseño (CONICET-UTN), Avellaneda 3657, S3002GJC Santa Fe, Argentina

[*] Corresponding author: mussati@santafe-conicet.gov.ar

Abstract

This work deals with the development and implementation of mathematical models in the General Algebraic Modeling System (GAMS) environment for optimization purposes, involving extrinsic functions that are executed outside GAMS from dynamic-link libraries (DLL) implemented in the programming language C. Three DLL libraries are developed to calculate thermodynamic properties: the Raoult's law for vapor-liquid equilibrium, the Non-Random Two-Liquid (NRTL) model, and the Peng-Robinson equation of state. A detailed description on how GAMS and DLL libraries interact is presented. Case studies dealing with the optimal design of multi-component distillation columns with increasing complexity levels are discussed. For the proposed case studies, the obtained results show that the usage of the proposed extrinsic functions allows to significantly enhance the model implementation compared to the traditional model implementation approach, and to considerably reduce the model size as well as the computational time required by the optimization algorithms.

Keywords: Mathematical Programming; Algebraic Modeling Languages; GAMS; Extrinsic Functions; External Equations; Distillation.

1. Introduction

The mathematical modeling of any chemical engineering process is a complex and challenging task because of the non-linear nature of the equations required to describe both the pieces of equipment and the process itself. Currently, algebraic modeling languages (AML), which are high-level computer programming languages, are widely employed to solve highly non-linear and large scale optimization problems, such as GAMS (General Algebraic Modeling System) (GAMS Development Corp., 2018a), AIMMS (Advanced Interactive Multidimensional Modeling System) (AIMMS B.V., 2018), AMPL (A Mathematical Programming Language) (AMPL Optimization Inc., 2018), FICO Xpress (Fair Isaac Corporation, 2018).

In chemical engineering, the estimation of physicochemical properties is a critical task to build models to simulate and/or to optimize a production process and it is one of the main sources of nonlinearities. The use of advanced physicochemical property estimation packages in algebraic modeling languages generally requires defining numerous parameters, variables, and intermediate equations that significantly increase the model's size (equations and variables). In addition, a realistic description of reaction kinetics, vapor-liquid-equilibrium, and/or caloric properties involves non-linear constraints leading to very complex models that often face convergence problems. To overcome this, several authors used external routines with the main aim of transferring calculation procedures to an external module (Tolsma et al., 2000, 2002; Poth et al., 2003). Tolsma et al. (2002) presented source-to-source code transformation techniques to properly incorporate external code into an equation-oriented process modeling environment. They employed Fortran to implement the external routines and ABACUSS II (Tolsma et al., 2000) as the equation-oriented modeling environment. The authors highlighted that the proposed techniques can be successfully used for incorporating external procedures into modular simulators for steady-state simulation and optimization.

External routines have been widely suggested to solve complex sub-problems in several process optimizations (Kravanja and Grossmann, 1996; Noronha et al., 1997). Also, external equilibrium calculations were implemented using process simulators (Caballero and Grossmann, 2010). Generally, the use of external modules in equation-based optimization problems exploits the nature of a subset of equality constraints that are part of the feasible region, to solve the problem more efficiently. This allows the use of tailored algorithms or even black-box type models.

In this work, GAMS (GAMS Development Corp., 2018a) is selected as the algebraic modeling language not only because it is widely used in chemical process optimization problems (Arias et al., 2016; Barttfeld et al., 2004, 2003; Manassaldi et al., 2016, 2014; Mores et al., 2018; Mussati et al., 2008; Onishi et al., 2017) but also because it allows creating an external calculation module. Besides the possibility of using in GAMS a "conventional" syntax in the model declaration for manipulating data as well as for relating constraints (with different types of variables: integer, binary, continuous), the GAMS Function Library Facility allows users to import functions from an external library into a GAMS model. Specifically, GAMS offers the possibility of integrating external functions packaged as a DLL – dynamic-link library – (if the operating system is Windows) or SO – shared object – (if the operating system is Unix).

GAMS provides two different ways to include external modules: *external equations* and *extrinsic functions* (GAMS Development Corp., 2018b), which were introduced in 1996 and 2011, respectively. They differentiate in several aspects, mainly in their usage and implementation way.

The extrinsic functions are more intuitive to use and also easier to implement since the single functions handle only the endogenous variables required for their computation and not the whole set of variables that needs to be transferred simultaneously as in the case of the external equation. In addition, extrinsic functions supplied by a DLL library can be used much more flexible. Furthermore, they can provide first and second order derivative information and are supported by NLP and MINLP solvers, while external equations provide first order derivative information only. On the other side, the external equations allow for the simultaneous solution of a whole equation system with dedicated algorithms, while the extrinsic functions are limited to the computation of scalar values and a maximum number of arguments of 20. Lastusilta et al. (2012) presented a comparative analysis between the use of external equations and extrinsic functions in several optimization problems. They concluded that the usage of extrinsic functions seems to be more intuitive, both options have different benefits, and there is not a clearly superior approach.

Several authors made the most of external equation feature in GAMS. Brusis (2003) employed external equations to estimate the thermodynamic properties and the column size for optimizing azeotropic distillation processes. Poth et al. (2003) further extended this approach to reactive distillation processes by using external equations to describe reaction kinetics. The authors concluded that the model's convergence behavior when using external equations in the model building resulted in significantly improvements with a lower number of iterations than traditional approach. In a similar way, Kossack et al. (2006) and Kraemer et al. (2009) addressed the optimal design of distillation columns by employing external equations in GAMS to calculate the thermodynamic properties (liquid activity coefficients and enthalpies) as well as the required derivatives. In Kossack et al. (2006), a combination of shortcut calculations using the rectification body method (RBM) and rigorous optimization is proposed. The former RBM method is used to identify feasible products by minimizing the energy demand required for the separation. The obtained solution is then used not only to initialize and bound the variables of the rigorous model. In the same way, Skiborowski et al. (2015) recently presented an approach for the optimization-based design of heterogeneous azeotropic distillation processes based on equilibrium tray models and rigorous thermodynamic models, employing GAMS. In order to guarantee a correct selection of the number of phases in each column tray, they proposed a phase stability test and reformulated the equilibrium equations by using external equations. The authors encapsulated the whole VLE or VLLE model and, therefore, performed the solution of an implicit function defined by a single or more sets of nonlinear equations. Some applications on membrane-assisted distillation processes, dividing wall columns, and shortcut modeling can be found in Skiborowski et al. (2018, 2014). Recker et al. (2015) proposed a systematic optimization-based approach, implemented in GAMS, for the design of chemical reaction-separation processes. They used external equations to calculate the vapor-liquid equilibrium (VLE) of the specific analyzed case. Waltermann and Skiborowski (2016) presented an efficient optimization-based method for the evaluation of different distillation configurations, also implemented in GAMS. The proposed approach is based on a superstructure equilibrium tray model considering rigorous thermodynamic models. Similarly to Recker et al. (2015), the thermodynamic properties are calculated by means of external equations. Schilling et al. (2017a, 2017b) simultaneously optimized the process and the working fluid of an Organic Rankine Cycle (ORC) using the Perturbed-Chain Statistical Associating Fluid Theory (PC-SAFT) equation of state for modeling the thermodynamic properties of the working fluid. The PC-SAFT and the process models were linked to GAMS using external equations. A significant contribution was presented by Bongarts and Mitsos (2017), who extended the use of implicit functions in the context of global optimization of process flowsheets which requires not only the evaluation of function values and derivative information, but also the propagation of relaxations.

Recently, Manassaldi (2017) proposed an optimization approach using extrinsic functions to obtain the optimal configuration and operating conditions of a Combined Cycle Gas Turbine (CCGT) for different target levels of electricity generation. Specifically, a DLL library with several extrinsic functions was created not only to calculate the physicochemical properties of the working fluids (water and combustion exhaust gas) but also to describe the behavior of the main process units (gas turbine, combustor, steam turbines, among others). The extrinsic functions included in the DLL libraries developed by Manassaldi (2017) can be easily extended to different case studies (utility system plants and several NGCCs (Manassaldi et al., 2016, 2014), absorption refrigeration cycles (Mazzei et al., 2014), among others) without modifying the source codes.

Based on this, it is clear the benefit of providing a set of thermodynamic packages that can be included into rigorous chemical process simulation/optimization models. Thus, a main purpose of this paper is to present a collection of DLL libraries implemented in the C programming language to facilitate the integration of rigorous thermodynamic packages (the Raoult's law, the Non-Random Two-Liquid (NRTL) model, and the Peng-Robinson equation of state) into algebraic modeling languages, in particular into GAMS.

To the best of our knowledge, this type of integration using extrinsic functions has not been reported in the literature. The DLL libraries for both simulation and optimization purposes in different application fields will be available for the readers. A database with 430 components is included, which allows considering many mixtures of different components.

This paper is organized as follows. Section 2 describes the developed DLL libraries. Section 3 describes the successive steps that must be executed during the DLL library loading process.

Section 4 introduces how the developed DLL libraries can be included in optimization mathematical models. Section 5 shows the application of the proposed DLL libraries to optimize distillation-based separation processes. Finally, Section 6 presents the conclusions and future works.

2. Library description

In this work, the capability of extrinsic functions to establish direct relationships between thermodynamic properties (enthalpy, entropy, etc.) and the intensive variables (temperature, pressure, and composition) of the main process streams is exploited. The three general-purpose thermodynamic libraries were developed:

– *RaoultLaw.dll*: Ideal solution (liquid phase) + Ideal gas (vapor phase).

– *NRTLideal.dll*: NRTL activity coefficient (liquid phase) (Renon and Prausnitz, 1968) + Ideal gas (vapor phase).

– *PengRobinson.dll*: Peng Robinson equation of state (both phases) (Peng and Robinson, 1976).

Each DLL library contains a set of different extrinsic functions and each function corresponds to a thermodynamic property. All functions require temperature (K), pressure (bar), and component mole fractions as input arguments. The number of extrinsic functions and the input arguments of each function automatically vary according to the number of involved compounds (up to 18 compounds). If *n* is the number of compounds, there will be (*6 + 2n*) extrinsic functions and each one will have (*n + 2*) input arguments. For example, a binary mixture results in eight extrinsic functions, each one having four input arguments.

For the sake of generality, all libraries contain the same extrinsic functions (Table 1) but each one contains a different method to estimate the thermodynamic properties. For instance, according to Table 1, the extrinsic function named as *rho_liq* returns the value of density corresponding to a liquid mixture. So, if the *PengRobinson.dll* library is used, the calculation is done from the compressibility factor; but if the *RaoultLaw.dll* is used, it is estimated from the Hakinson-Thomson method. The estimation methods used in each DLL library are presented in Appendix A (Tables A.1–A.3).

Insert Table 1

The thermodynamic libraries were implemented in the C programming language using DevC++ (Bloodshed, 2018) as a development environment (IDE) and TDM-GCC (TDM-GCC, 2018) as a compiler. As mentioned, the number of functions and input arguments change with the number of compounds. For a better performance, all the extrinsic functions involve analytic implementations of the corresponding gradient vectors and Hessian matrixes.

3. DLL library loading process

Each DLL library was implemented in such a way to be adapted to different situations since the appropriate thermodynamic estimates depend on the type of mixture to be modeled. Therefore, the users must choose the most suitable available library for the mixture under study.

Extrinsic functions are used like any traditional mathematical function of GAMS (intrinsic function). But, unlike intrinsic functions, they need to be 'pre-loaded' for their usage. Therefore, a series of statements should be introduced at the beginning of the GAMS file. Figure 1 illustrates the successive steps that must be executed during the DLL library loading process.

Insert Figure 1

The set of developed libraries automatically identifies the involved compounds by reading a TXT file that is created by the user. For this purpose, in Step 1 the involved compounds are defined in a TXT file according to their ID number in the pure compound database (Kooijman and Taylor, 2016). In this step, the binary interaction parameters must also be provided for the *PengRobinson.dll* and *RaoultLaw.dll* libraries. To avoid identification problems, each library has an assigned file name that the user must respect. More information about the creation of the TXT files can be found in Manassaldi et al. (2018).

In Step 2 the DLL library, which contains the desired extrinsic functions to be used in the GAMS model, is called. Steps 1 and 2 are included as statements at the beginning of the GAMS file.

Steps 3 and 4 are automatically performed by the DLL library. In Step 3 the DLL library reads the TXT file (that contains the compound IDs) and determines the number of involved compounds in the mixture. The name of the TXT file has to be respected so that the library can identify it. Once the number of compounds n is known, the number of extrinsic functions ($6 + 2n$) and input arguments of each one ($2 + n$) are established. It is important to note that the libraries automatically adapt to the number of compounds, so the source code must not be modified whatever mixture is to be modeled (with the components included in the library). Extrinsic functions allow storing information in the memory to be used during the solver execution. So, in Step 4, once all the compounds are identified, the database is accessed and the parameters of each compound are stored (critical properties, heat capacity polynomial constants, etc.). In addition, from the information contained in the TXT file, binary interaction parameters can also be stored (if necessary).

In Step 5 the extrinsic functions that are going to be used in the model are defined. In this step it is not necessary to include all available functions but only those of interest. This step is also included as a statement within the GAMS file. Then, once the previous steps have been successfully completed, the extrinsic functions are already available for their use. According to the user needs, they can be employed to define parameters, initialize model variables, and/or explicitly include them in equations.

4. DLL library inclusion in optimization mathematical models

As mentioned earlier, the model implementation using extrinsic functions allows a simple and friendly inclusion of the thermodynamic package. This approach can be extended in a general way to more complex processes with the higher number of variables and equations.

Figure 2 presents two approaches to build a chemical process mathematical model in Algebraic Modeling Language software. The solid line strategy follows the classical implementation way, which is hereafter named as MS1. The dotted line strategy uses the thermodynamic packages implemented by extrinsic functions, which is hereafter named as MS2. As is observed, both methodologies only differ on how the thermodynamic properties are modeled and implemented.

Insert Figure 2

The first step in both alternatives (*Step 1*) is to define the system under study (real world); for instance, it can be an entire chemical process or individual pieces of process equipment. Then, a mathematical model including sets of equations and inequations is developed to describe all the existing physical and chemical phenomena (*Step 2*). This step corresponds to the development of a theoretical model that describes the process (or piece of equipment) under study.

For implementation convenience, the mathematical model was separated into two main groups of equations (*Steps 3* and *4*). As is seen in *Step 3*, the first group of equations is the same in both approaches. It contains the process characteristic equations, where the most frequent are, for instance, the mass and energy balances, equilibrium equations, cost estimation, and process restrictions.

The second group of equations corresponds to the calculation of thermodynamic properties (thermodynamic package). As illustrated, *Step 4** (in the 'traditional' formulation) includes all the equations and necessary variables for the calculation of thermodynamic properties. While *Step 4*** (in the proposed approach) only includes direct functions instead of a set of mathematical constraints.

The use of extrinsic functions takes advantage of the direct relationships between thermodynamic properties and main variables (T, P, and composition); this corresponds to a partition of the set of constraints. As mentioned, the extrinsic functions must be pre-loaded for their use following the steps described in Section 3.

5. Case studies

The developed DLL libraries are here employed to optimize distillation-based separation processes as illustrative cases. Even though the distillation processes have been studied for decades and there exist a vast amount of published papers on design, operating modes, and control strategies of such processes, they offer an excellent benchmark for testing novel solution strategies due to their nonlinear characteristics and the trade-offs existing among model variables (Malinen, 2011). Indeed, as mentioned in Section 1, several authors utilized and tested external equation in the modeling of this type of separation processes (Brusis, 2003; Poth et al., 2003; Skiborowski et al., 2015).

Figure 3 illustrates a schematic of a simple distillation column. The mixture (F) is fed at the feed tray (FT), dividing the column in two main sections: a) an enriching or rectification section, where volatile components are removed by contacting the rising vapor stream with the down-flowing liquid stream; and b) a stripping section, where the heavier components in the liquid phase are concentrated. The reboiler (REB) is a heat exchanger where a vapor stream is generated, which moves up the column. The liquid stream leaving the reboiler is the bottom product (B). At the top of the column, a total condenser (COND) is considered to condense the hot vapor leaving the column. A fraction of the condensate is recycled back to the top of the column (reflux R) and the other fraction is the top product or distillate (D).

Insert Figure 3

The optimal design of distillation columns involves several trade-offs among the energy consumption in the reboiler, the number of distillation trays (sizing), and product specifications, among others. For instance, for a given product purity, the higher number of trays (capital cost), the lower reflux ratio and the lower energy consumption in the reboiler (energy cost).

To illustrate the usage of the DLL libraries, two case studies consisting on the optimization of distillation columns with different complexity levels are presented. These case studies were selected as examples of classical modeling problems to show the modeling strategy here presented and its performance. Here, the focus is mainly on the modeling task following the guidelines provided above (Figure 2).

5.1 Case study 1

The case study 1 consists in separating by means of distillation an ethanol-water mixture into pure water and a mixture at the azeotropic conditions.

The optimization problem can be stated as follows.

Given a mixture of 1000 kmol/h of ethanol (A) and 1000 kmol/h of water (B) at atmospheric pressure (saturated condition) and the following target specifications: a minimum water separation of 60% with a bottom product purity of 99%, the problem is to determine the optimal number of column trays, the feed tray location, the heat transfer areas of the condenser and reboiler and their corresponding heat loads, by minimization of the total annual cost, which is calculated in terms of the annualized investments and annual operating costs.

To this end, a superstructure-based model similar to that proposed by (Yeomans and Grossmann (2000) is used (Figure 4a). For this case study, the superstructure considers a minimum of ten existing (or fixed) stages (from *s10* to *s19*), i.e. they always will be part of the optimal solution and will not be removed. The only reason to fix this minimum number of trays is to reduce the number of binary variables involved, and thus to reduce the calculation time.

Insert Figure 4 (4a and 4b)

As shown in Figure 4a, the proposed superstructure contains a maximum number of 20 trays, of which 8 (*s2* to *s9*) are modeled as conditional trays (Figure 4b). This means that, depending on the optimization criterion, all or some of these trays can be removed by the optimization algorithm. Similar to the remaining trays (*s10* to *s19*), the condenser (*s1*) and the reboiler (*s20*) are assumed as existing trays. In addition, the trays *s2* to *s19* are 'candidate' to be the feeding tray.

For modeling purpose, the following three main assumptions are considered: a) vapor phase behaves ideally, b) the NRTL model is appropriate for estimating the liquid phase activity coefficients (*NRTLideal.dll* library), and c) the separation takes place at atmospheric pressure.

By applying the methodology proposed in Section 4, the equations corresponding to the process (Step 3 in Fig. 2) and the set of constraints that correspond to the physicochemical package (Step 4 in Fig. 2) are presented next.

Equation (1) corresponds to the component mass balances in the condenser (stage *s1*). *L*, *V*, and *D* refer to the liquid, vapor, and distillate molar flow rates, respectively; *x* and *y* refer to the liquid and vapor phase molar fractions, respectively. *s* refers to the column stages (*s = s1 to s20*) and *i* to the components (A and B). The subset COND(*s*) represents the condenser stage (*s1*).

$$V_{s+1} y_{s+1,i} = \left(L_s + D\right) x_{s,i} \qquad \forall i; \forall s / s \in COND\left(s\right) \qquad (1)$$

The energy balance is given by Eq. (2):

$$V_{s+1} H_{s+1}^{v} = Q^{cond} + \left(L_s + D\right) H_s^{l} \qquad \forall s / s \in COND\left(s\right) \qquad (2)$$

where Q^{cond} refers to the heat duty in the condenser, and H^l and H^v to the enthalpy of the liquid and vapor phases, respectively.

Equations (3) and (4) are the mass and energy balances in the reboiler (stage *s20*), respectively:

$$L_{s-1}x_{s-1,i} = V_s y_{s,i} + L_s x_{s,i} \qquad \forall i; \forall s / s \in REB(s) \tag{3}$$

$$L_{s-1}H^l_{s-1} + Q^{reb} = V_s H^v_s + L_s H^l_s \qquad \forall s / s \in REB(s) \tag{4}$$

where Q^{reb} refers to the heat duty required in the reboiler. The subset REB(s) represents the reboiler stage (*s20*).

The feed stream (F_s) can be placed in one of the intermediate stages. Equations (5) and (6) are the corresponding mass and energy balances. The subset TRAY(s) includes all intermediate stages (*s2* to *s19*).

$$F_s xf_i + L_{s-1}x_{s-1,i} + V_{s+1}y_{s+1,i} = V_s y_{s,i} + L_s x_{s,i} \qquad \forall i; \forall s / s \in TRAY(s) \tag{5}$$

$$F_s Hf + L_{s-1}H^l_{s-1} + V_{s+1}H^v_{s+1} = V_s H^v_s + L_s H^l_s \qquad \forall i; \forall s / s \in TRAY(s) \tag{6}$$

where xf_i and Hf are model parameters and they refer to the composition and enthalpy of the main feed stream, respectively. The mass balance in the splitter that distributes the main feed stream to the candidate feed trays is given by Eq. (7):

$$LF = \sum_{s \in TRAY(s)} F_s \tag{7}$$

The following composition constraints are imposed to the liquid and vapor phases (Eq. (8) and (9), respectively):

$$\sum_i x_{s,i} = 1 \qquad \forall s \tag{8}$$

$$\sum_i y_{s,i} = 1 \qquad \forall s \tag{9}$$

As mentioned, the objective of this process is to separate the water content of the feed stream to increase the ethanol concentration in the distillate stream. So, a minimum water purity of 0.99 in the bottom product and a separation of the inlet water content higher than 60% are imposed through Eq. (10) and (11), respectively:

$$x_{s,A} \geq 0.99 \qquad \forall s / s \in REB(s) \tag{10}$$

$$L_s x_{s,A} \geq 0.6 LF\, xl_A \qquad \forall s / s \in REB(s) \tag{11}$$

Discrete variables are used to model the presence or absence of trays. As shown in Fig. 4a and 4b, series of conditional trays (NOFIXED(*s*)) and fixed trays (FIXED(*s*)) are proposed.

In each fixed stage (condenser, reboiler, and fixed trays), the liquid and vapor phases are in equilibrium. Therefore, the fugacity of the component *i* in the liquid phase ($fug^{l}_{s,i}$) must be equal to the fugacity of the same component in the vapor phase ($fug^{v}_{s,i}$) (Eq. 12):

$$fug^{l}_{s,i} = fug^{v}_{s,i} \qquad \forall i; \forall s / s \in \left(FIXED(s) \cup COND(s) \cup REB(s) \right) \tag{12}$$

In the same way, since both phases (liquid and vapor) are in equilibrium, their temperatures are the same (Eq. 13):

$$T^{l}_{s} = T^{v}_{s} \qquad \forall s / s \in \left(FIXED(s) \cup COND(s) \cup REB(s) \right) \tag{13}$$

The diameter of a tray (*TD*) is calculated as follows:

$$TD_{s}^{\,2} = 0.77072 V_{s} \sqrt{\left(\sum y_{s,i} MW_{i} \right) / \rho^{v}_{s}} \quad \forall s / s \in TRAY(s) \tag{14}$$

where *MW* is the component molecular weight and ρ^{v} the vapor molar density. Then, the column diameter (*CD*) must be equal to or greater than the diameter of each tray (Eq. 15):

$$CD \geq TD_{s} \qquad \forall s / s \in TRAY(s) \tag{15}$$

The Boolean variable N_s in Eq. 16 defines the existence of a candidate tray in the optimal solution. If N_s is true the tray *s* is selected and the vapor-liquid equilibrium equations are considered. Otherwise, if N_s is false the tray is removed; so, feeding is forbidden and the liquid phase pass through the stage without any change in composition and temperature. Thus, if a tray is removed no mass transfer takes place.

$$\begin{bmatrix} N_{s} \\ fug^{l}_{s,i} = fug^{v}_{s,i} \ \forall i \\ T^{l}_{s} = T^{v}_{s} \\ \\ \\ \end{bmatrix} \vee \begin{bmatrix} \neg N_{s} \\ x_{s,i} = x_{s-1,i} \ \forall i \\ L_{s} = L_{s-1} \\ T^{l}_{s} = T^{l}_{s-1} \\ F_{s} = 0 \end{bmatrix} \qquad \forall s / s \in NOFIXED(s) \tag{16}$$

The total number of trays (*NT*) is the sum of both fixed and candidate trays (Eq. 17):

$$NT = \sum_{s \in FIXED(s)} 1 + \sum_{s \in NOFIXED(s)} n_{s} \tag{17}$$

where n_s is the binary variable associated to the Boolean variable N_s.

The objective function consists in minimizing the total annual cost (*TAC*), which accounts for the annualized capital expenditure (*annCAPEX*) and the operating expenditure (*OPEX*):

$$TAC = annCAPEX + OPEX \tag{18}$$

The *annCAPEX* takes into account the cost for column tray (C_{tray}), column shell (C_{shell}), condenser (C_{cond}), and reboiler (C_{reb}), which are calculated as follows (in \$/year):

$$C_{shell} = 3458.9 \cdot NT \cdot CD_{col}{}^{1.066} \cdot H_{tray}{}^{0.802} \tag{19}$$

$$C_{tray} = 430.45 \cdot NT \cdot CD^{1.55} H_{tray} \tag{20}$$

$$C_{reb} = 167.97\left(Q^{reb}\right)^{0.65} \tag{21}$$

$$C_{cond} = 446.51\left(Q^{cond}\right)^{0.65} \tag{22}$$

where H_{tray} is the tray height.

The *OPEX* is calculated in terms of the cooling and steam utility costs (Eq. 23):

$$OPEX = \mathrm{C}_{\mathrm{steam}} \cdot M_{steam} + C_{cw} \cdot M_{cw} \tag{23}$$

where M_{steam} and M_{cw} are the requirements of steam and cooling water, respectively, expressed in t/year. The associated specific costs are C_{steam}=10.02 \$/t and C_{cw}= 0.09 \$/t.

After defining the process's characteristic equations (Eq. (1) to (23)), it is required to select the thermodynamic packages to be used. As explained in Section 4, two implementation ways of the thermodynamic packages are considered (Table 2). The former implements all thermodynamic equations in the traditional way (MS1). While the second strategy involves extrinsic functions from the *NRTLideal.dll* library (MS2). As indicated in Table 3, both ways are coupled with Eq. (1) to (23).

It is observed in Table 2 that Eq. (24) to (27) in MS1 and Eq. (28) to (31) in MS2 are used to calculate the thermodynamic properties (fugacities and enthalpies) required for mass and energy balances and equilibrium equations.

In MS1 alternative, the activity coefficient (γ), saturation pressure (P^{sat}), and pointing factor (*poy*) are needed to estimate the liquid phase component fugacity (Eq. (24)). On the other hand, the ideal gas enthalpy of pure component (H^{IG}), heat of vaporization of pure component (ΔH^{vap}), and component excess enthalpy (ΔH^{ex}) are needed to estimate the liquid and vapor phase enthalpies (Eq. (26) and (27)). Equations (B.1) to (B.22) in Appendix B are necessary to calculate all mentioned variables involved in Eq. (24) to (27).

Insert Table 2

When extrinsic functions are employed (MS2), the main set of the thermodynamic equations express the direct relationship between the desired property (fugacity and enthalpy) and the main process stream variables (temperature, pressure, and composition). Thus, the optimization problems can be mathematically expressed as is shown in Table 3:

Insert Table 3

Both problems were implemented in GAMS. The discrete decisions used to select the number of trays lead to mixed-integer nonlinear programming (MINLP) models, which are solved using the standard Branch and Bound algorithm (SBB) as MINLP solver and CONOPT as nonlinear programming (NLP) solver for the intermediate nodes. Table 4 compares the model sizes, CPU times, number of iterations, number of explored nodes, and the corresponding total annual cost.

Insert Table 4

Table 4 shows considerable differences in the model sizes. When the DLL library is used (MS2 implementation), the number of equations is reduced by 68% (from 1478 to 477) and the continuous variables by 73% (from 1375 to 374) when compared to the traditional strategy MS1. The computation time for both models is low; however, when the extrinsic functions are used, the time is reduced by 42% (from 4.137 s to 2.396 s). The number of iterations required by the MS1 implementation is 30% less than by the MS2 implementation (826 vs. 1139). Both models obtained the same optimal solution, which is presented in Fig. 5.

Insert Figure 5

The optimal solution consists on a distillation column with 12 trays (6 trays were removed) that is fed 5 stages above the reboiler (at s=15), with a reflux ratio of 0.3883 and a bottom product flow rate of 606.06 kmol/h.

In order to verify the obtained results, the optimal solution was compared with solutions obtained from several process simulators. To this end, the degrees of freedom of the simulated distillation column were fixed using the optimal output values obtained by the GAMS model. Table 5 compares the average differences obtained in each case.

Insert Table 5

As shown, the MINLP output values are in agreement with those obtained with process simulators. The very small differences – 0.493% in the worst case – are mainly due to the pure compound database incorporated in each process simulator and some model assumptions.

5.2 Case study 2

The mathematical model used in the previous case study was properly extended to solve the process configuration consisting of two coupled distillation columns (Fig. 6) to treat a mixture consisting of three components. Both columns are coupled by means of stream mixers and splitters. Similarly to the case study 1, the number of fixed trays in both columns was appropriately selected to reduce the model size.

Insert Figure 6

Precisely, a saturated liquid mixture of n-pentane (A), n-hexane (B), and n-heptane (C) is fed at a flow rate of 26 kmol/s with a molar fraction of A, B, and C of 0.33, 0.33, and 0.34, respectively. A product purity of 0.98 and a minimum recovery of 98% are specified for the three components. The separation takes place at atmospheric pressure. Based on the mixture type, the Peng-Robinson EOS is used to estimate the thermodynamic properties. Therefore, the *PengRobinson.dll* library is selected.

The optimization problem is similar to the previous one but with increased complexity from the computational cost point of view since it involves a higher number of trade-offs and, consequently, the number of variables and equations significantly increases accordingly. The following discrete and continuous decisions are provided as a result of the optimization model (Fig. 7):

– Number of trays in each column.
– Feed tray location of each column.
– Operation parameters (operation conditions) of each column.
– Distillation sequence (for example: AB|C and A|B).
– Total annual cost of the distillation sequence.
– Heat transfer area required by the condenser and reboiler of each column.

Also, a comparison of the performance between both MS1 and MS2 modeling strategies is presented in Table 6.

Insert Table 6

As is seen in Table 6, a significant reduction in the model size is obtained when the DLL library is used, which is by 75% in the number of equations (from 5910 to 1510) and 78% in the number of variables (from 5345 to 1145). The CPU time is reduced by more than half. Both strategies obtained the same solution (Fig. 7), with a total annual cost of 153410 $/year (Table 6).

As shown in Fig. 7, the optimal solution results in the following simple distillation sequence: the main feed stream is sent to the first column where the first component is separated (A|BC). The bottom product of the first column is sent to the second one where the remaining components are separated (B|C).

Insert Figure 7

5.3 Comparison of solutions obtained using external equations and extrinsic functions.

Finally, a comparison of the performance between the DLL libraries using extrinsic functions here developed and using external equations employed by other authors is performed through another example of distillation. A GAMS model code implemented by the research group in Process Systems Engineering AVT.PT–RWTH Aachen University (Aachener Verfahrenstechnik, 2019a) that employs a set of external equations is used for comparison purpose. The whole VLE or VLLE model is encapsulated in the GAMS code. The source code of the external equations is available for download and must be compiled by the user to create the corresponding DLL libraries. To perform the comparison, the GAMS model was downloaded from the published software collection (Aachener Verfahrenstechnik, 2019b) and the external equations were properly replaced by the proposed extrinsic functions, retaining the constraints associated to the mass and energy balances. The replacement of the external equations can be easily carried out due to the friendly implementation of the GAMS model performed by the authors.

Similarly to the previous case studies, the optimization problem consists in determining the optimal number of stages, the feed tray location, sizes, and operating conditions of the distillation column that minimize the total annual cost. The main design specifications are the following:

– The column can involve a maximum number of 80 stages. The model will determine the optimal number of stages.
– The feed stream is a mixture of methanol and ethanol at a molar flow rate of 50 kmol/s each, at 1.01325 bar (saturated condition).
– Purity specifications at the top and bottom streams are 0.995 and 0.0001 mole fraction of methanol, respectively.

Table 7 compares the results corresponding to both solutions.

Insert Table 7

As shown, the values of the objective functions differ only in about 2.5% (154647.6 €/year vs. 158532.2 €/year). The configuration and the column size are also similar. The small differences are due to the different theoretical models used to calculate the vapor-liquid equilibrium. The Wilson's method (a theoretical model based on activity coefficients) is used in the original model (Aachener Verfahrenstechnik, 2019b) and the NRTL method (*NRTLideal.dll*) in the current model. The computing time required by the usage of extrinsic functions is lower than of external equations but the number of iterations is slightly higher.

As mentioned, the external equations used in the original model from Aachener Verfahrenstechnik (2019b) to calculate the thermodynamic properties were replaced by those presented in this paper. Since no new variables were added to the model the number of variables is the same in both models, as shown in Table 7. However, the number of equations in the current model is lower than the original one. This is because the original libraries require *n+3* equations (*n* is the number of components) for each equilibrium stage while the current libraries require *n+2* equations. For this reason, the difference in the number of equations is equal to the number of stages (80).

In the original model, a series of intermediate models are previously solved to obtain a feasible initialization of the principal model. The model proposed in this paper (using extrinsic functions) solves the same intermediate problems but the computation times were enhanced in all cases (results not shown).

5. Conclusion

Different dynamic-link libraries with extrinsic functions for GAMS have been developed and implemented for the calculation of thermodynamic properties according to different theoretical approaches. Specifically, libraries for the Raoult's Law (*Raoultlaw.dll*), the Peng-Robinson equation of state (*PenRobinson.dll*), and the Non-Random Two-Liquid model (*NRTLideal.dll*) were developed.

Two case studies with different complexity levels were presented to illustrate the performance of the libraries. The first case study dealt with the optimal synthesis and design of a simple distillation column and the second one of a three-component simple distillation sequence. Both cases were compared with a traditional implementation of a model including the corresponding thermodynamic package. The results showed a significant decrease in both the model size and computation time. This decrease was also pointed out by Poth et al. (2003) and Skiborowski et al. (2015), who improved the convergence of GAMS models by employing external equations, transferring complex calculations to external modules.

In this work, the generalization of physicochemical packages (libraries) was possible by using extrinsic functions. The developed libraries can be easily included in mass and energy balances of different process-units. Due to their generality, they can be easily applied to simulate and/or optimize any type of chemical processes. It is clear that the use of the introduced libraries based on extrinsic functions to calculate physicochemical properties greatly facilitates the modeling task. In our opinion, this is the main contribution of the paper.

The files with the developed DLL libraries are available in the contributed software section of the GAMS website (GAMS Development Corp., 2018c).

In future works, DLL libraries for (a) the UNIversal QUAsi-Chemical (UNIQUAC) activity model, (b) the Wilson's activity model, (c) the Soave-Redlich-Kwong (SRK) equation of state, (d) the modified Benedict-Webb-Rubin (mBWR) equation of state, and (e) the Perturbed-Chain Statistical Associating Fluid Theory (PC-SAFT) equation of state will be also developed and available to users. Also, the convenience of using extrinsic functions over embedding the thermodynamic models in an external equation will be investigated.

Another challenge is the creation of external modules for process equipment representation. The use of external functions will facilitate the implementation of conventional models (equation-oriented models) or non-conventional models (black-box or neural-type models) in mathematical optimization problems.

These challenges will be applied to our previous models such as integrated combined cycles and CO2 capture plants (Mores et al., 2018), seawater desalination processes including single purpose plants (Mussati et al., 2003b, 2001) as well as dual purpose plants (Mussati et al., 2005, 2004, 2003a), among others.

Acknowledgements

The financial support from the Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET) and the Facultad Regional Rosario of the Universidad Tecnológica Nacional from Argentina are gratefully acknowledged.

Appendix A. Thermodynamic property estimation methods

Tables A.1 and A.2 show the methods for estimating thermodynamic properties implemented in each extrinsic function according to the different developed libraries. As mentioned, all the libraries contain the same extrinsic functions but they differ in the used theoretical model. More information about the theoretical models implemented in this article can be found in Poling et al. (2001).

Insert Table A.1

Insert Table A.2

As can be noted in Tables A.1 and A.2, the only difference between the *NRTLideal.dll* and *RaoultLaw.dll* libraries is that the latter assumes that the liquid phase behaves ideally. That is, *RaoultLaw.dll* assumes a unitary activity coefficient and a unitary poynting factor, and neglects the enthalpy and entropy excesses. All the extrinsic functions included in the *NRTLideal.dll* and *Raoultlaw.dll* libraries follow a direct calculation sequence. That is, once the input values for the extrinsic functions (e.g. pressure, temperature, and component composition) are provided, no iterative process is needed to calculate the output value (e.g. enthalpy).

Unlike the two previous DLL libraries, the *PengRobinson.dll* library has an intermediate iterative resolution process. The cubic equation of state is solved iteratively within the external calculation routine. The implemented sequence uses a specific strategy for solving cubic equations of state, as proposed by Deiters and Macías-Salinas (2014). In case of not being able to find the root of the cubic polynomial an error is reported and the GAMS solver ends the process. Table A.3 lists the extrinsic functions involved in the *PengRobinson.dll* library.

Insert Table A.3

As illustration, extracts of the source codes for computing the gradient vector and Hessian matrix in the extrinsic function corresponding to the liquid enthalpy in the Peng-Robinson library (PengRobinson.DLL) are provided in the supplementary material associated to this work.

All the developed libraries have the same pure compound database. All parameters and mathematical functions corresponding to the thermodynamic properties of the pure compounds were extracted from the Chemsep 7.15 database (Kooijman and Taylor, 2016) and are presented in Table A.4.

Insert Table A.4

Appendix B. NRTL equations

Equations (B.1) to (B.22) are related to the MS1 strategy presented in the Case Study 1. They correspond to the NRTL activity coefficient model and enthalpy estimation. A detailed theoretical description of Eq. (B.1) to (B.22) can be found in Poling et al. (2001).

$$\log\left(P_{s,i}^{sat}\right)=A_i+\frac{B_i}{T_s^l}+C_i\log\left(T_s^l\right)+D_i\left(T_s^l\right)^{E_i}\quad \forall s;\forall i \tag{B.1}$$

$$Tr_{s,i}=T_s^l/Tc_i\quad \forall s;\forall i \tag{B.2}$$

$$V_{s,i}^{(0)}=1+a\left(1-Tr_{s,i}\right)^{\frac{1}{3}}+b\left(1-Tr_{s,i}\right)^{\frac{2}{3}}+c\left(1-Tr_{s,i}\right)+d\left(1-Tr_{s,i}\right)^{\frac{4}{3}}\qquad \forall s;\forall i \tag{B.3}$$

$$V_{s,i}^{(\delta)}=\left(e+fTr_{s,i}+gTr_{s,i}{}^{2}+hTr_{s,i}{}^{3}\right)/\left(Tr_{s,i}-1.00001\right)\quad \forall s;\forall i \tag{B.4}$$

$$V_{s,i}=V_i^{*}V_{s,i}^{(0)}\left[1-\omega_{SRK,i}V_{s,i}^{(\delta)}\right]\quad \forall s;\forall i \tag{B.5}$$

$$poy_{s,i}=\left(V_{s,i}\exp\left(P_s-P_{s,i}^{sat}\right)\right)/\left(RT_s^l\right)\quad \forall s;\forall i \tag{B.6}$$

$$\tau_{s,i,j}=a_{i,j}/\left(RT_s^l\right)\qquad \forall s;\forall i;\forall j \tag{B.7}$$

$$G_{s,i,j}=\exp\left(-\alpha_{i,j}\tau_{s,i,j}\right)\qquad \forall s;\forall i;\forall j \tag{B.8}$$

$$S_{s,i}=\sum_{j=1}^{n}x_{s,j}G_{s,j,i}\qquad \forall s;\forall i \tag{B.9}$$

$$C_{s,i}=\sum_{j=1}^{n}x_{s,j}G_{s,j,i}\tau_{s,j,i}\quad \forall s;\forall i \tag{B.10}$$

$$\log\gamma_{s,i}=\frac{C_{s,i}}{S_{s,i}}+\sum_{k=1}^{n}x_{s,k}G_{s,i,k}\left(\frac{\tau_{s,i,k}}{S_{s,k}}-\frac{C_{s,k}}{S_{s,k}{}^{2}}\right)\qquad \forall s;\forall i \tag{B.11}$$

$$\Delta H_{s,i}^{vap}=A_i\left(1-Tr_{s,i}\right)^{B_i+C_iTr_{s,i}+D_iTr_{s,i}{}^{2}+C_iTr_{s,i}{}^{3}}\quad \forall s;\forall i \tag{B.12}$$

$$\left(\frac{\partial \tau}{\partial T}\right)_{s,i,j} = -a_{i,j} \Big/ \left(R \left(T_s^l\right)^2 \right) \qquad \forall s; \forall i; \forall j \tag{B.13}$$

$$\left(\frac{\partial G}{\partial T}\right)_{s,i,j} = -\alpha_{i,j} G_{s,i,j} \left(\frac{\partial \tau}{\partial T}\right)_{s,i,j} \qquad \forall s; \forall i; \forall j \tag{B.14}$$

$$\left(\frac{\partial S}{\partial T}\right)_{s,i} = \sum_{j=1}^{n} x_{s,j} \left(\frac{\partial G}{\partial T}\right)_{s,j,i} \qquad \forall s; \forall i \tag{B.15}$$

$$\left(\frac{\partial C}{\partial T}\right)_{s,i} = \sum_{j=1}^{n} x_{s,j} \left(\left(\frac{\partial G}{\partial T}\right)_{s,j,i} \tau_{s,j,i} + G_{s,j,i} \left(\frac{\partial \tau}{\partial T}\right)_{s,j,i} \right) \qquad \forall s; \forall i \tag{B.16}$$

$$M1_{s,i} = \sum_{k=1}^{n} x_{s,k} \left(\left(\frac{\partial G}{\partial T}\right)_{s,i,k} \tau_{s,i,k} + G_{s,i,k} \left(\frac{\partial \tau}{\partial T}\right)_{s,i,k} \right) \Big/ S_{s,k} \qquad \forall s; \forall i \tag{B.17}$$

$$M2_{s,i} = \sum_{k=1}^{n} -x_{s,k} \left(\left(\frac{\partial G}{\partial T}\right)_{s,i,k} C_{s,k} + G_{s,i,k} \left(\frac{\partial C}{\partial T}\right)_{s,k} + G_{s,i,k} \tau_{s,i,k} \left(\frac{\partial S}{\partial T}\right)_{s,k} \right) \Big/ \left(S_{s,k}\right)^2 \qquad \forall s; \forall i \tag{B.18}$$

$$M3_{s,i} = \sum_{k=1}^{n} 2 x_{s,k} G_{s,i,k} C_{s,k} \left(\frac{\partial S}{\partial T}\right)_{s,k} \Big/ \left(S_{s,k}\right)^3 \qquad \forall s; \forall i \tag{B.19}$$

$$\left(\frac{\partial \log \gamma}{\partial T}\right)_{s,i} = \left(\frac{\partial C}{\partial T}\right)_{s,i} \frac{1}{S_{s,i}} - \left(\frac{\partial S}{\partial T}\right)_{s,i} \frac{C_{s,i}}{S_{s,i}^{\;2}} + M1_{s,i} + M2_{s,i} + M3_{s,i} \qquad \forall s; \forall i \tag{B.20}$$

$$\Delta H_{s,i}^{ex} = -R \left(T_s^l\right)^2 \left(\frac{\partial \log \gamma}{\partial T}\right)_{s,i} \qquad \forall s; \forall i \tag{B.21}$$

$$H_{s,i}^{p,IG} = \Delta H_{i,f}^{0} + \int_{298.15}^{T_s^p} cp_i^{IG} \, dT \qquad \forall s; \forall i; \forall p \tag{B.22}$$

References

Aachener Verfahrenstechnik, 2019a. Process Systems Engineering (PT) - RWTH AACHEN UNIVERSITY Aachener Verfahrenstechnik. http://www.avt.rwth-aachen.de/go/id/ioaf/lidx/1 (accessed 23 January 2019).

Aachener Verfahrenstechnik, 2019b. Process synthesis software collection download area. http://www.avt.rwth-aachen.de/cms/AVT/Forschung/Software/Softwaresammlung-Prozesssynthese/~ipyh/Downloadbereich/?lidx=1 (accessed 23 January 2019).

AIMMS B.V., 2018. Advanced Interactive Multidimensional Modeling System (AIMMS). https://www.AIMMS.com (accessed 7 January 2019).

AMPL Optimization Inc., 2018. A Mathematical Programming Language (AMPL). https://ampl.com/ (accessed 7 January 2019).

Arias, A.M., Mussati, M.C., Mores, P.L., Scenna, N.J., Caballero, J.A., Mussati, S.F., 2016. Optimization of multi-stage membrane systems for CO2 capture from flue gas. Int. J. Greenh. Gas Control 53, 371–390. https://doi.org/10.1016/j.ijggc.2016.08.005

Barttfeld, M., Aguirre, P.A., Grossmann, I.E., 2004. A decomposition method for synthesizing complex column configurations using tray-by-tray GDP models. Comput. Chem. Eng. 28, 2165–2188. https://doi.org/10.1016/j.compchemeng.2004.03.006

Barttfeld, M., Aguirre, P.A., Grossmann, I.E., 2003. Alternative representations and formulations for the economic optimization of multicomponent distillation columns. Comput. Chem. Eng. 27, 363–383. https://doi.org/10.1016/S0098-1354(02)00213-2

Bloodshed, 2018. Dev-C++. https://www.bloodshed.net/devcpp.html (accessed 7 January 2019).

Bongartz, D., Mitsos, A., 2017. Deterministic global optimization of process flowsheets in a reduced space using McCormick relaxations. J. Glob. Optim. 69, 761–796. https://doi.org/10.1007/s10898-017-0547-4

Brusis, D., 2003. Synthesis and optimisation of distillation processes with MINLP techniques. München, Techn. Univ., Diss.

Caballero, J.A., Grossmann, I.E., 2010. Hybrid Simulation-Optimization Algorithms for Distillation Design, in: Pierucci, S., Ferraris, G.B. (Eds.), Computer Aided Chemical Engineering, 20 European Symposium on Computer Aided Process Engineering. Elsevier, pp. 637–642. https://doi.org/10.1016/S1570-7946(10)28107-5

Deiters, U.K., Macías-Salinas, R., 2014. Calculation of Densities from Cubic Equations of State: Revisited. Ind. Eng. Chem. Res. 53, 2529–2536. https://doi.org/10.1021/ie4038664

Fair Isaac Corporation, 2018. FICO® Xpress Optimization. https://www.fico.com/en/products/fico-xpress-optimization (accessed 7 January 2019).

GAMS Development Corp., 2018a. General Algebraic Modeling System (GAMS). https://www.gams.com/ (accessed 7 January 2019).

GAMS Development Corp., 2018b. The GAMS User’s Guide. https://www.gams.com/latest/docs/UG_MAIN.html (accessed 7 January 2019).

GAMS Development Corp., 2018c. GAMS - Contributed Software. https://www.gams.com/community/contributed-software/ (accessed 7 January 2019).

Kooijman, H., Taylor, R., 2016. ChemSep v7.15 pure component data.

Kossack, S., Kraemer, K., Marquardt, W., 2006. Efficient Optimization-Based Design of Distillation Columns for Homogenous Azeotropic Mixtures. Ind. Eng. Chem. Res. 45, 8492–8502. https://doi.org/10.1021/ie060117h

Kraemer, K., Kossack, S., Marquardt, W., 2009. Efficient Optimization-Based Design of Distillation Processes for Homogeneous Azeotropic Mixtures. Ind. Eng. Chem. Res. 48, 6749–6764. https://doi.org/10.1021/ie900143e

Kravanja, Z., Grossmann, I.E., 1996. A Computational Approach for the Modeling/Decomposition Strategy in the MINLP Optimization of Process Flowsheets with Implicit Models. Ind. Eng. Chem. Res. 35, 2065–2070. https://doi.org/10.1021/ie950424f

Lastusilta, T., Bussieck, M., Emet, S., 2012. Extrinsic functions in GAMS. Presented at the International Conference on Operations Research, Hannover. https://old.gams.com/presentations/or2012ef.pdf

Malinen, I., 2011. Improving the robustness with modified bounded homotopies and problem-tailored solving procedures (Ph. D.). University of Oulu, Oulu, Finlandia.

Manassaldi, J.I., 2017. Optimal synthesis and design of chemical processes involving discrete and continuous decisions using mathematical programming (PhD Thesis). National Technological University, Regional Faculty of Cordoba, Argentina.

Manassaldi, J.I., Arias, A.M., Scenna, N.J., Mussati, M.C., Mussati, S.F., 2016. A discrete and continuous mathematical model for the optimal synthesis and design of dual pressure heat recovery steam generators coupled to two steam turbines. Energy 103, 807–823. https://doi.org/10.1016/j.energy.2016.02.129

Manassaldi, J.I., Mores, P.L., Scenna, N.J., Mussati, S.F., 2014. Optimal Design and Operating Conditions of an Integrated Plant Using a Natural Gas Combined Cycle and Postcombustion CO2 Capture. Ind. Eng. Chem. Res. 53, 17026–17042. https://doi.org/10.1021/ie5004637

Manassaldi, J.I., Mussati, M.C., Scenna, N.J., Mussati, S.F., 2018. User's manual to use thermodynamic libraries in GAMS through extrinsic functions. GAMS.

Mazzei, M.S., Mussati, M.C., Mussati, S.F., 2014. NLP model-based optimal design of LiBr–H2O absorption refrigeration systems. Int. J. Refrig. 38, 58–70. https://doi.org/10.1016/j.ijrefrig.2013.10.012

Mores, P.L., Manassaldi, J.I., Scenna, N.J., Caballero, J.A., Mussati, M.C., Mussati, S.F., 2018. Optimization of the design, operating conditions, and coupling configuration of combined cycle power plants and CO2 capture processes by minimizing the mitigation cost. Chem. Eng. J. 331, 870–894. https://doi.org/10.1016/j.cej.2017.08.111

Mussati, S.F., Aguirre, P., Scenna, N., 2003a. Dual-purpose desalination plants. Part II. Optimal configuration. Desalination 153, 185–189. https://doi.org/10.1016/S0011-9164(02)01126-8

Mussati, S.F., Aguirre, P., Scenna, N.J., 2001. Optimal MSF plant design. Desalination, European Conference on DESALINATION AND THE ENVIRONMENT WATER SHORTAGE 138, 341–347. https://doi.org/10.1016/S0011-9164(01)00283-1

Mussati, S.F., Aguirre, P.A., Scenna, N.J., 2005. Optimization of alternative structures of integrated power and desalination plants. Desalination, Desalination and the Environment 182, 123–129. https://doi.org/10.1016/j.desal.2005.03.012

Mussati, S.F., Aguirre, P.A., Scenna, N.J., 2004. A rigorous, mixed-integer, nonlineal programming model (MINLP) for synthesis and optimal operation of cogeneration seawater desalination plants. Desalination, Desalination Strategies in South Mediterranean Countries 166, 339–345. https://doi.org/10.1016/j.desal.2004.06.088

Mussati, S.F., Aguirre, P.A., Scenna, N.J., 2003b. Novel Configuration for a Multistage Flash-Mixer Desalination System. Ind. Eng. Chem. Res. 42, 4828–4839. https://doi.org/10.1021/ie020318v

Mussati, S.F., Barttfeld, M., Aguirre, P.A., Scenna, N.J., 2008. A disjunctive programming model for superstructure optimization of power and desalting plants. Desalination 222, 457–465. https://doi.org/10.1016/j.desal.2007.01.162

Noronha, S., Gruhn, G., Kravanja, Z., 1997. Handling implicit model formulations in MINLP optimization. Comput. Chem. Eng., Supplement to Computers and Chemical Engineering 21, S499–S504. https://doi.org/10.1016/S0098-1354(97)87551-5

Onishi, V.C., Ravagnani, M.A.S.S., Jiménez, L., Caballero, J.A., 2017. Multi-objective synthesis of work and heat exchange networks: Optimal balance between economic and environmental performance. Energy Convers. Manag. 140, 192–202. https://doi.org/10.1016/j.enconman.2017.02.074

Peng, D.-Y., Robinson, D.B., 1976. A New Two-Constant Equation of State. Ind. Eng. Chem. Fundam. 15, 59–64. https://doi.org/10.1021/i160057a011

Poling, B.E., Prausnitz, J.M., O'Connell, J.P., 2001. The Properties of Gases and Liquids, 5th ed. McGraw-Hill Education, New York.

Poth, N., Brusis, D., Stichlmair, J., 2003. Rigorous optimization of reactive distillation in GAMS with the use of external functions, in: Kraslawski, A., Turunen, I. (Eds.), Computer Aided Chemical Engineering, European Symposium on Computer Aided Process Engineering-13. Elsevier, pp. 869–874. https://doi.org/10.1016/S1570-7946(03)80226-2

Recker, S., Skiborowski, M., Redepenning, C., Marquardt, W., 2015. A unifying framework for optimization-based design of integrated reaction–separation processes. Comput. Chem. Eng., Special Issue: Selected papers from the 8th International Symposium on the Foundations of

Computer-Aided Process Design (FOCAPD 2014), July 13-17, 2014, Cle Elum, Washington, USA 81, 260–271. https://doi.org/10.1016/j.compchemeng.2015.03.014

Renon, H., Prausnitz, J.M., 1968. Local compositions in thermodynamic excess functions for liquid mixtures. AIChE J. 14, 135–144. https://doi.org/10.1002/aic.690140124

Schilling, J., Gross, J., Bardow, A., 2017a. Integrated design of ORC process and working fluid using process flowsheeting software and PC-SAFT. Energy Procedia 129, 129–136. https://doi.org/10.1016/j.egypro.2017.09.184

Schilling, J., Tillmanns, D., Lampe, M., Hopp, M., Gross, J., Bardow, A., 2017b. Integrating working fluid design into the thermo-economic design of ORC processes using PC-SAFT. Energy Procedia 129, 121–128. https://doi.org/10.1016/j.egypro.2017.09.179

Skiborowski, M., Harwardt, A., Marquardt, W., 2015. Efficient optimization-based design for the separation of heterogeneous azeotropic mixtures. Comput. Chem. Eng., A Tribute to Ignacio E. Grossmann 72, 34–51. https://doi.org/10.1016/j.compchemeng.2014.03.012

Skiborowski, M., Recker, S., Marquardt, W., 2018. Shortcut-based optimization of distillation-based processes by a novel reformulation of the feed angle method. Chem. Eng. Res. Des. 132, 135–148. https://doi.org/10.1016/j.cherd.2018.01.019

Skiborowski, M., Wessel, J., Marquardt, W., 2014. Efficient Optimization-Based Design of Membrane-Assisted Distillation Processes. Ind. Eng. Chem. Res. 53, 15698–15717. https://doi.org/10.1021/ie502482b

TDM-GCC, 2018. A compiler suite for 32- and 64-bit Windows based on the GNU toolchain. http://tdm-gcc.tdragon.net/ (accessed 7 January 2019).

Tolsma, J.E., Clabaugh, J.A., Barton, P.I., 2002. Symbolic Incorporation of External Procedures into Process Modeling Environments. Ind. Eng. Chem. Res. 41, 3867–3876. https://doi.org/10.1021/ie0107946

Tolsma, J.E., Clabaugh, J.A., Barton, P.I., 2000. ABACUSS II: Advanced Modeling Environment and Embedded Process Simulator. Camb. MA. http://yoric.mit.edu/abacuss2/abacuss2.html

Waltermann, T., Skiborowski, M., 2016. Efficient optimization-based design of energetically intensified distillation processes, in: Kravanja, Z., Bogataj, M. (Eds.), Computer Aided Chemical Engineering, 26 European Symposium on Computer Aided Process Engineering. Elsevier, pp. 571–576. https://doi.org/10.1016/B978-0-444-63428-3.50100-4

Yeomans, H., Grossmann, I.E., 2000. Disjunctive Programming Models for the Optimal Design of Distillation Columns and Separation Sequences. Ind. Eng. Chem. Res. 39, 1637–1648. https://doi.org/10.1021/ie9906520

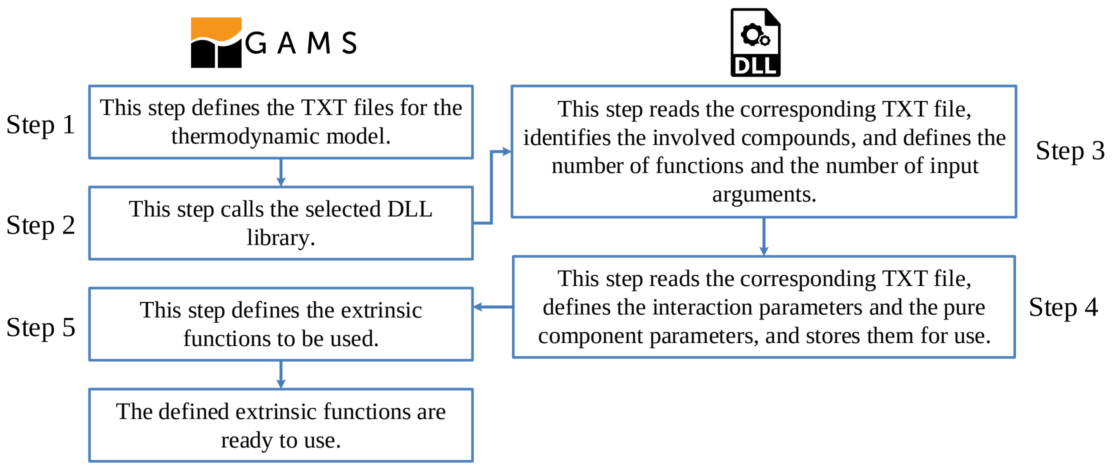

GAMS
DLL
Step 1
This step defines the TXT files for the thermodynamic model.
Step 2
This step calls the selected DLL library.
Step 3
This step reads the corresponding TXT file, identifies the involved compounds, and defines the number of functions and the number of input arguments.
Step 4
This step reads the corresponding TXT file, defines the interaction parameters and the pure component parameters, and stores them for use.
Step 5
This step defines the extrinsic functions to be used.
The defined extrinsic functions are ready to use.

Figure 1. Steps of the DLL library loading process.

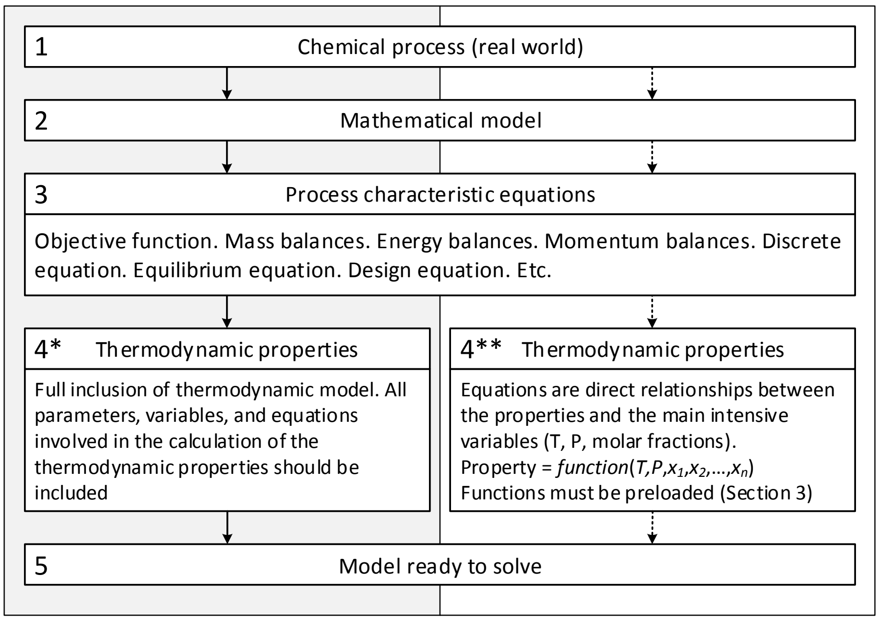

1 Chemical process (real world)
2 Mathematical model
3 Process characteristic equations
Objective function. Mass balances. Energy balances. Momentum balances. Discrete equation. Equilibrium equation. Design equation. Etc.
4* Thermodynamic properties
Full inclusion of thermodynamic model. All parameters, variables, and equations involved in the calculation of the thermodynamic properties should be included
4** Thermodynamic properties
Equations are direct relationships between the properties and the main intensive variables (T, P, molar fractions).
Property = function(T,P,x1,x2,...,xn)
Functions must be preloaded (Section 3)
5 Model ready to solve

Figure 2. Classical strategy (solid line) and DLL-based strategy (dotted line) to build mathematical models in a AML software.

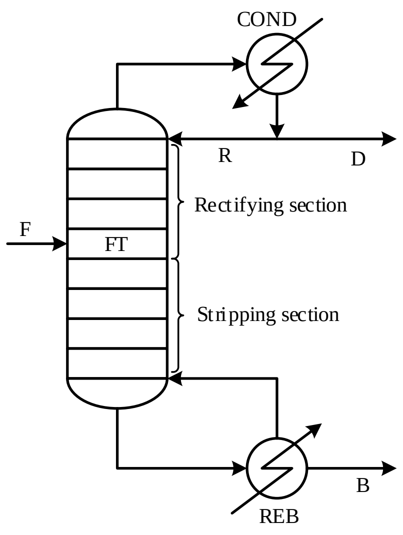


Figure 3. Schematic of a distillation column.

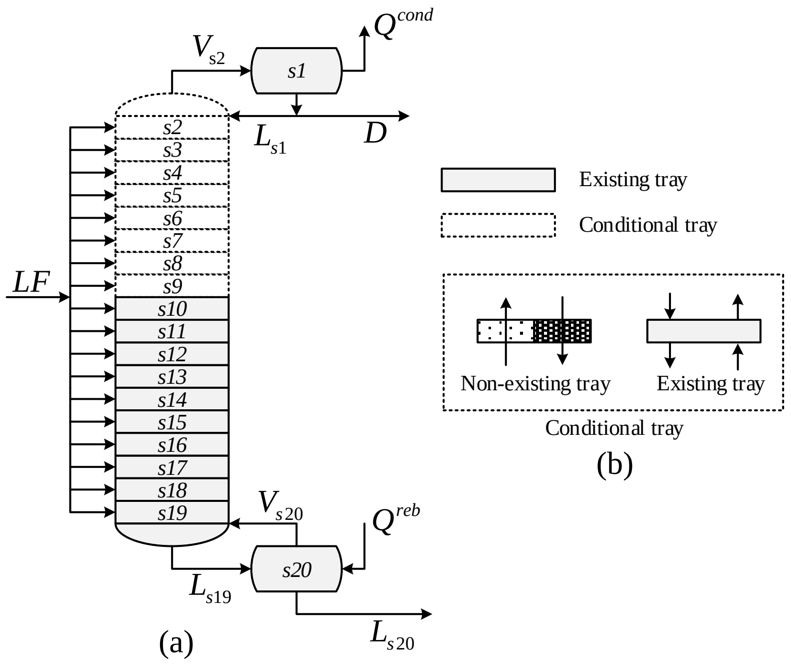


Figure 4. Schematics of (a) distillation column superstructure and (b) conditional tray approach, for modeling purpose.

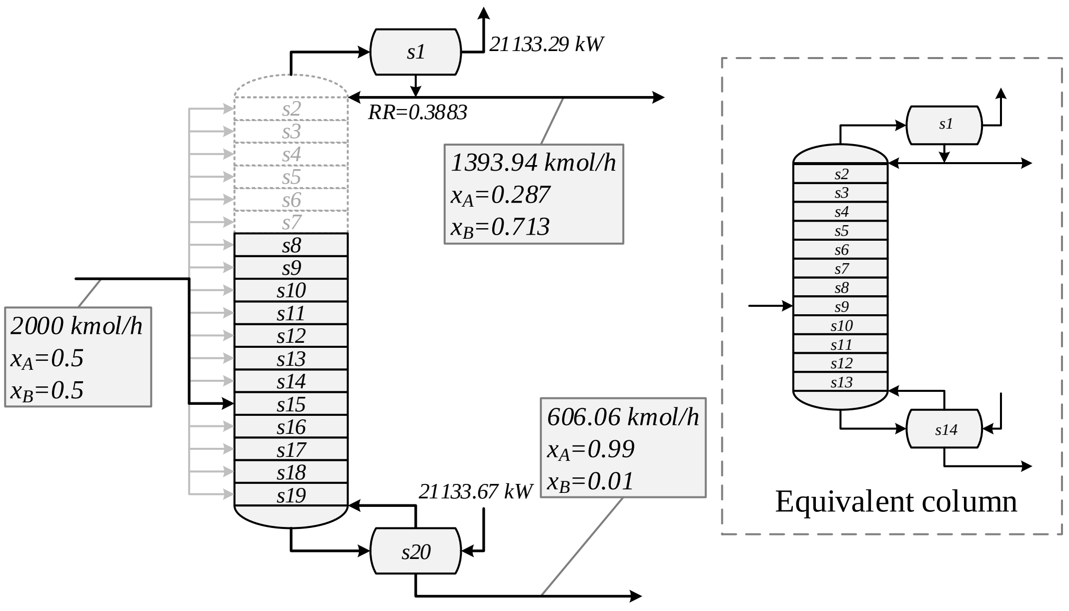


Figure 5. Optimal configuration of the distillation column obtained for the case study 1.

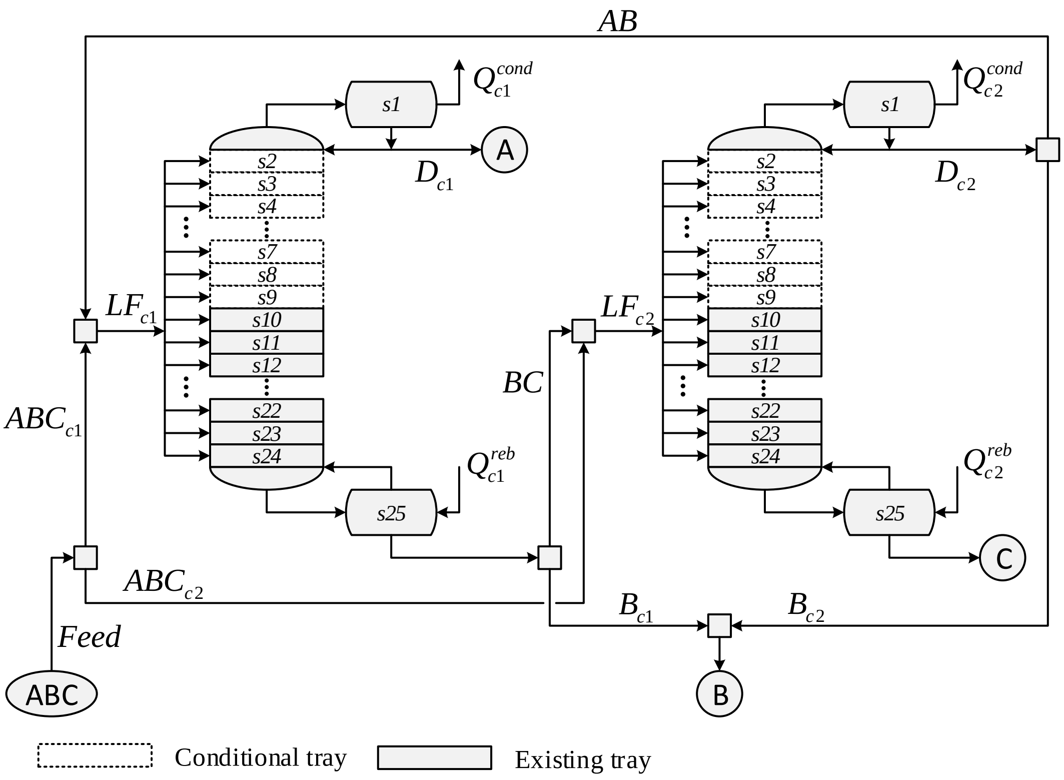


Figure 6. Schematic of a distillation sequence superstructure for a mixture of three components.

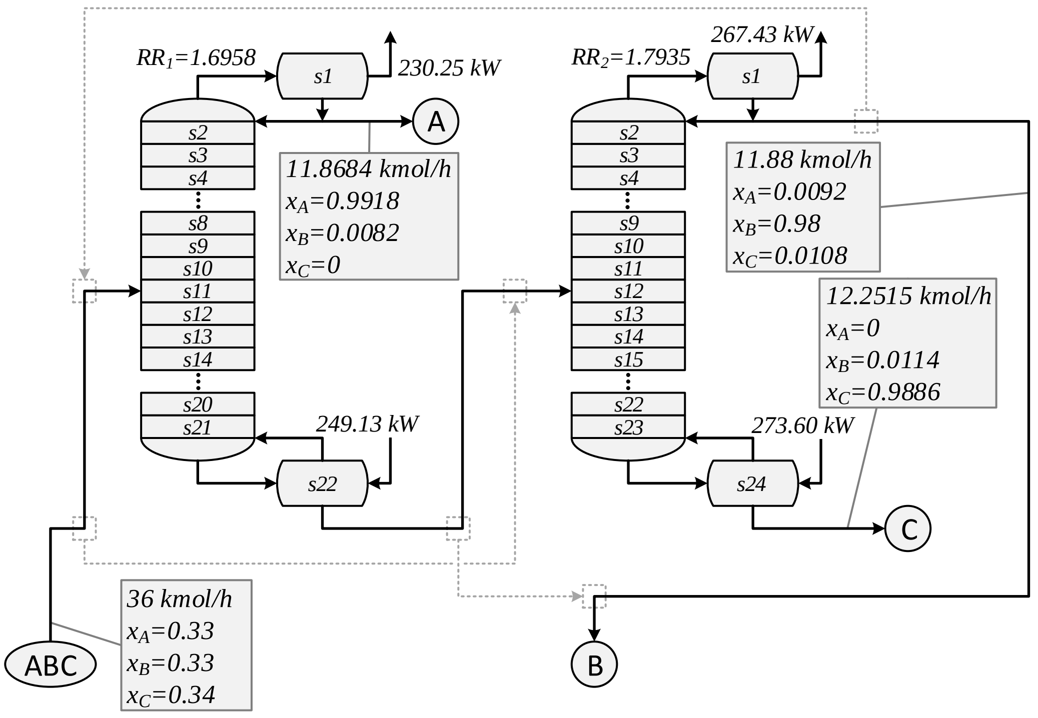


Figure 7. Optimal distillation sequence obtained for the case study 2.

Figure captions

Figure 1. Steps of the DLL library loading process.

Figure 2. Classical strategy (solid line) and DLL-based strategy (dotted line) to build mathematical models in a AML software.

Figure 3. Schematic of a distillation column.

Figure 4. Schematics of (a) distillation column superstructure and (b) conditional tray approach, for modeling purpose.

Figure 5. Optimal configuration of the distillation column obtained for the case study 1.

Figure 6. Schematic of a distillation sequence superstructure for a mixture of three components.

Figure 7. Optimal distillation sequence obtained for the case study 2.

Table 1. Extrinsic functions considered in each DLL library.

Property	Comments	Units
rho_liq(T, P, x_1, x_2, ..., x_n)	Density of the liquid phase	mol/m^3
rho_vap(T, P, x_1, x_2, ..., x_n)	Density of the vapor phase	mol/m^3
h_liq(T, P, x_1, x_2, ..., x_n)	Enthalpy of the liquid phase	J/mol
h_vap(T, P, x_1, x_2, ..., x_n)	Enthalpy of the vapor phase	J/mol
s_liq(T, P, x_1, x_2, ..., x_n)	Entropy of the liquid phase	J/(mol·K)
s_vap(T, P, x_1, x_2, ..., x_n)	Entropy of the vapor phase	J/(mol·K)
f1_liq(T, P, x_1, x_2, ..., x_n)	Fugacity of component *1* in the liquid phase	bar
f1_vap(T, P, x_1, x_2, ..., x_n)	Fugacity of component *1* in the vapor phase	bar
f2_liq(T, P, x_1, x_2, ..., x_n)	Fugacity of component *2* in the liquid phase	bar
f2_vap(T, P, x_1, x_2, ..., x_n)	Fugacity of component *2* in the vapor phase	bar
fn_liq(T, P, x_1, x_2, ..., x_n)	Fugacity of component *n* in the liquid phase	bar
fn_vap(T, P, x_1, x_2, ..., x_n)	Fugacity of component *n* in the vapor phase	bar

Table 2. Two implementation ways of the thermodynamic packages.

Classical approach (MS1)		DLL library-based strategy (MS2)	
$fug^{l}_{s,i} = \gamma_{s,i} x_{s,i} P^{sat}_{s,i}\, poy_{s,i} \quad \forall s; \forall i$	(24)	$fug^{l}_{s,i} = f^{l}_{i}\left(T^{l}_{s}, P_s, x_{s,A}, x_{s,B}\right) \quad \forall s; \forall i$	(28)
$fug^{v}_{s,i} = y_{s,i} P_s \quad \forall s; \forall i$	(25)	$fug^{v}_{s,i} = f^{v}_{i}\left(T^{v}_{s}, P_s, y_{s,A}, y_{s,B}\right) \quad \forall s; \forall i$	(29)
$H^{l}_{s} = \sum_{i=1}^{n} x_{s,i}\left(H^{l,IG}_{s,i} - \Delta H^{vap}_{s,i} + \Delta H^{ex}_{s,i}\right) \quad \forall s$	(26)	$H^{l}_{s} = h^{l}\left(T^{l}_{s}, P_s, x_{s,A}, x_{s,B}\right) \quad \forall s$	(30)
$H^{v}_{s} = \sum_{i=1}^{n} y_{s,i} H^{v,IG}_{s,i} \quad \forall s$	(27)	$H^{v}_{s} = h^{v}\left(T^{v}_{s}, P_s, y_{s,A}, y_{s,B}\right) \quad \forall s$	(31)
Equations (B.1) to (B.22) (Appendix B)			

Table 3. Optimization mathematical models considering both strategies (MS1 and MS2).

Classical strategy (MS1)	DLL library based strategy (MS2)

Minimize TAC
Subject to:
Eq.(1) to Eq.(23)
Eq.(24) to Eq.(27)
Eq.(B.1) to Eq.(B.22)
	Minimize TAC
	Subject to:
	Eq.(1) to Eq.(23)
	Eq.(28) to Eq.(31)

Table 4. Model statistics of case study 1.

	Classical strategy (MS1)	DLL library based strategy (MS2)
Number of equations	1478	477
Continuous variables	1375	374
Discrete variables	8	8
Time for NLPs (s)	4.137	2.396
Number of iterations	826	1139
B&B nodes	16	16
TAC ($/year)	2982000	2982000

Table 5. Deviation of average output values obtained by the GAMS model with DLL libraries (MS2) and process simulators.

	Temperature	Liquid flow	Vapor flow
ChemSep (ChemSep, 2018)	< 0.0008%	< 0.0002%	< 0.0002%
Hysys (Aspentech, 2018)	0.047%	0.364%	0.185%
Dwsim (Medeiros, 2018)	0.016%	0.493%	0.260%

Table 6. Model statistics of case study 2.

	Classical strategy (MS1)	DLL library-based strategy (MS2)
Number of equations	5910	1510
Continuous variables	5345	1145
Discrete variables	16	16
Time for NLPs (s)	410.813	193.348
Number of iterations	19120	34562
B&B nodes	160	160
TAC ($/year)	153410	153410

Table 7. Performance comparison between external equations and extrinsic functions.

	External equations libraries (Aachener Verfahrenstechnik, 2019b).	Extrinsic function libraries (This work)
Total annual cost (€/year)	154647.6	158532.2
Number of equilibrium stages	70	72
Feed stage location	32	26
Column height (m)	38	39
Column diameter (m)	0.646	0.65
Resource usage (s)	4.57	0.219
Number of iterations	654	665
Single variable	1542	1542
Single equation	2104	2024
NLP solver	SNOPT 7.2-12.1	SNOPT 7.2-12.1

Table A.1. Methods for estimating thermodynamic properties in the DLL library corresponding to the NRTL model (*NRTLideal.dll)*

Function	**Calculation method**
rho_liq	Liquid mixture density by the Hankinson and Thomson method.
rho_vap	Ideal gas density
h_liq	Ideal liquid enthalpy plus excess enthalpy (from activity coefficient).
h_vap	Ideal gas enthalpy.
s_liq	Ideal liquid entropy plus excess entropy (from activity coefficient).
s_vap	Ideal gas entropy.
f#_liq	Liquid fugacity from activity coefficient (NRTL model) including the Poynting factor.
f#_vap	Ideal gas fugacity.

Table A.2. Methods for estimating thermodynamic properties in the DLL library corresponding to the Raoult's law (*RaoultLaw.dll*)

Function	Calculation method
rho_liq	Liquid mixture density by the Hankinson and Thomson method.
rho_vap	Ideal gas density.
h_liq	Ideal liquid enthalpy.
h_vap	Ideal gas enthalpy.
s_liq	Ideal liquid entropy.
s_vap	Ideal gas entropy.
f#_liq	Liquid fugacity considering unitary activity coefficient without including the Poynting factor.
f#_vap	Ideal gas fugacity.

Table A.3. Methods for estimating thermodynamic properties in the DLL library corresponding to the Peng-Robinson EOS (*PengRobinson.dll*)

Function	Calculation method
rho_liq	Liquid density estimated using the liquid phase compressibility factor.
rho_vap	Vapor density estimated using the vapor phase compressibility factor.
h_liq	Liquid enthalpy estimated using the ideal gas enthalpy and the liquid departure enthalpy.
h_vap	Vapor enthalpy estimated using the ideal gas enthalpy and the vapor departure enthalpy.
s_liq	Liquid entropy estimated using the ideal gas entropy and the liquid departure entropy.
s_vap	Vapor entropy estimated using the ideal gas entropy and the vapor departure entropy.
f#_liq	Liquid fugacity estimated using the liquid fugacity coefficient.
f#_vap	Vapor fugacity estimated using the vapor fugacity coefficient.

Table A.4. Thermodynamic property estimation methods for pure compounds.

Property	Estimation method
Ideal gas heat capacity	Reid-Prausnitz-Poling (RPP) fourth order polynomial.
Heat of vaporization	Using a temperature correlation obtained from the Chemsep database.
Vapor pressure	Using a temperature correlation obtained from the Chemsep database.
Saturated liquid volume	Hankinson and Thomson method.
Critical properties	Chemsep database.

Supplementary material

Development of Extrinsic Functions for Optimal Synthesis and Design—Application to Distillation-based Separation Processes

***Juan I. Manassaldi* [a], *Miguel C. Mussati* [a,b], *Nicolás J. Scenna* [a], *Sergio F. Mussati* [a,b]**

[a] CAIMI Centro de Aplicaciones Informáticas y Modelado en Ingeniería (UTN-FRRo), Zeballos 1341, S2000BQA, Rosario, Argentina
[b] INGAR Instituto de Desarrollo y Diseño (CONICET-UTN), Avellaneda 3657, S3002GJC, Santa Fe, Argentina
jmanassaldi@frro.utn.edu.ar, mmussati@santafe-conicet.gov.ar, nscenna@santafe-conicet.gov.ar, mussati@santafe-conicet.gov.ar

1. Introduction

This supplementary material provides to all the users the basic steps to use three general-purpose thermodynamic libraries in GAMS employing extrinsic functions.

- RaoultLaw.dll: Ideal solution (liquid phase) + Ideal gas (vapor phase).
- NRTLideal.dll: NRTL activity coefficient (liquid phase) + Ideal gas (vapor phase).
- PengRobinson.dll: Peng Robinson equation of state (both phases).

Below, general characteristics of the developed libraries are briefly summarized:

- Contain a database of 430 pure compounds.
- In a txt file, the IDs of the desired compounds and their interaction parameters (if necessary) should be defined.
- All functions have as input arguments the Temperature, Pressure and molar fraction of each mixture component. For example, for a binary mixture, the functions will have 4 input arguments.
- The input arguments of the function vary with the number of compounds involved.
- They support up to 18 compounds. Temperature + Pressure + 18 compounds = 20 argument (maximum arguments of extrinsic function for GAMS).
- All extrinsic functions have an analytic implementation of their gradient vector and Hessian matrix.
- Extrinsic Functions implemented in each library:
  - Liquid and vapor phase density.
  - Liquid and vapor phase enthalpy.
  - Liquid and vapor phase entropy.
  - Fugacity of each component in each phase (vapor and liquid).
- The database for pure compounds is taken from:

  ChemSep v7.15 pure component data - Copyright (c) Harry Kooijman and Ross Taylor (2016) - http://www.perlfoundation.org/artistic_license_2_0
- The libraries were developed in *Dev C ++* and using *tdm-gcc* as a compiler.
- More information about the library can be found in the compilation section of the *lst* file.

2. Basic steps required for configure and use the libraries

2.1. Compounds Assignment

Before including any of the libraries, the desired compounds must be assigned in a *txt* file. As shown in Table 1, each library has a defined file name.

Table 1. ID filename for each library

Library	ID file name
RaoultLaw.dll	*RaoultLawID.txt*
PengRobinson.dll	*PengRobinsonID.txt*
NRTLideal.dll	*NRTLidealID.txt*

Then, the compounds desired by the users are defined from their ID in the pure compounds database (ChemSep v7.15 pure component data - Copyright (c) Harry Kooijman and Ross Taylor). Appendix 1 presents a list including the available compounds with their corresponding IDs.

An illustrative example considering the Peng Robinson's equation of state and five compounds (propane, isobutane, n-butane, isopentane and n-pentane) is shown below:

```
$onecho > PengRobinsonID.txt
ID1 3
ID2 4
ID3 5
ID4 8
ID5 7
$offecho
```

As indicated, only one space should be used to separate the compound number and its database ID.

2.2. Interaction parameter definition

Depending on the selected thermodynamic package, it is necessary to define a group of interaction parameters. Again, each library has a file name assigned to each interaction group, as shown in Table 2.

Table 2. Interaction parameters file name for each library

Library	Interaction parameters file name	Units
PengRobinson.dll	*PengRobinsonaij.txt*	unitless
NRTLideal.dll	*NRTLidealaij.txt*	cal/mol
	NRTLidealalphaij.txt	unitless

Only binary interaction parameters that are not repeated should be defined. Interaction parameters declaration for the previously defined mixture is shown below.

```
$onecho > PengRobinsonaij.txt
a12 -0.0078
a13 0.0033
a14 0.0111
```

```
a15 0.0267
a23 -0.0004
a24 0.0005043
a25 0.00067951
a34 0.00021669
a35 0.0174
a45 0.06
$offecho
```

As example, the definition of parameters corresponding to a mixture consisting of ethanol and water using the *NRTLideal.dll* library is shown below.

```
$onecho > NRTLidealID.txt
ID1 1921
ID2 1102
$offecho
$onecho > NRTLidealaij.txt
a12 -57.9601
a21 1241.7396
$offecho
$onecho > NRTLidealalphaij.txt
alpha12 0.2937
$offecho
```

As shown (for *NRTLideal.dll*), the values of the parameters a_{12} and a_{21} must be defined because they are different. But, $alpha_{12}$ and $alpha_{21}$ have the same values. Therefore, $alpha_{12}$ is only defined and then the library assigns internally the same values for $alpha_{21}$. In the current version, the dependence of the interaction parameters with the temperature is neglected. In future versions, this will be added to improve the NRTL library capabilities.

2.3. Including the developed libraries in GAMS

The following internal coding is used to include the libraries into GAMS:

```
$FuncLibIn <InternalLibName> <ExternalLibName>
```

For example, the *NRTLideal.dll* library is included as follows:

```
$FuncLibIn NRTLideal NRTLideal.dll
```

The *NRTLideal.dll* file must be placed in the subdirectory *gamsdir/projdir*, otherwise, the corresponding fullpath must be specified. The library must be included after the definition of the compounds and interaction parameters.

Once the library is included, the functions arguments are automatically assigned to specify temperature, pressure, and compositions. Thus, the total number of arguments required depends on the number of compounds. For instance, the arguments needed for a binary mixture are four (T, P, x_1, x_2). In the compilation section in the *lst* file, it is possible to check if the compounds have been well identified. For example, the following information corresponds to *NRTLideal.dll* library execution (ethanol and water mixture).

```
FUNCLIBIN  NRTLideal NRTLideal.dll
Function Library NRTLideal
NRTL + IG Property Package v0.9 by Ph.D. J.I. Manassaldi (jmanassaldi@frro.utn.edu.ar); Ph.D. N.J.
Scenna; Ph.D. M.C. Mussati; Ph.D. S.F. Mussati (mussati@santafe-conicet.gov.ar)
GAMS Development Corporation

Mod. Function                          Description
Type

NLP  rho_liq(temperature [k],pressure [bar],water,ethanol)liquid phase molar density [mol/m3]
NLP  rho_vap(temperature [k],pressure [bar],water,ethanol)vapor phase molar density [mol/m3]
NLP  h_liq(temperature [k],pressure [bar],water,ethanol)liquid phase molar enthalpy [J/mol]
NLP  h_vap(temperature [k],pressure [bar],water,ethanol)vapor phase molar enthalpy [J/mol]
NLP  s_liq(temperature [k],pressure [bar],water,ethanol)liquid phase molar entropy [J/(mol.K)]
NLP  s_vap(temperature [k],pressure [bar],water,ethanol)vapor phase molar entropy [J/(mol.K)]
NLP  f1_liq(temperature [k],pressure [bar],water,ethanol)liquid phase fugacity of component 1 [bar]
NLP  f1_vap(temperature [k],pressure [bar],water,ethanol)vapor phase fugacity of component 1 [bar]
NLP  f2_liq(temperature [k],pressure [bar],water,ethanol)liquid phase fugacity of component 2 [bar]
NLP  f2_vap(temperature [k],pressure [bar],water,ethanol)vapor phase fugacity of component 2 [bar]
```

To avoid inconsistencies, it is important to observe the units of the input and output arguments of the functions.

2.4. Functions definition

After the library is included, the necessary functions must be defined. This task is also done using an internal coding of GAMS, which is indicated below:

```
function <InternalFuncName> /<InternalLibName>.<FuncName>/;
```

Thus, by applying the above internal code, the function for *liquid enthalpy* corresponding to the *NRTLideal.dll* library is defined as follows:

```
function hliq /NRTLideal.h_liq/;
```

In this example, for convenience, the original extrinsic function *h_liq* was redefined (for GAMS code) as *hliq*.

Appendix 2 shows the definition of the libraries considered for the case studies presented in this work.

3. Usage of libraries

Once the previous steps have been completed, the developed extrinsic functions are already available for use. They can be used to define parameters or include them in an equation.

In this section, an illustrative optimization example to show a detailed application of one of the developed libraries (*NRTLideal.dll*) is presented.

The objective of the optimization problem is to calculate the composition and temperature of a binary minimum-boiling homogeneous azeotrope. Precisely, a water-ethanol mixture is considered and the pressure is fixed at 1.0132 bar.

To do this, the following GAMS model has been used:

```
$onecho > NRTLidealID.txt
ID1 1921
ID2 1102
$offecho
$onecho > NRTLidealaij.txt
a12 1241.7396
a21 -57.9601
$offecho
$onecho > NRTLidealalphaij.txt
alpha12 0.2937
$offecho
$funclibin NRTLideal NRTLideal.dll
function f1l  /NRTLideal.f1_liq /;
function f2l  /NRTLideal.f2_liq /;
function f1v  /NRTLideal.f1_vap /;
function f2v  /NRTLideal.f2_vap /;

sets
i compounds /water,ethanol/
;
Parameter
P pressure [bar] /1.0132/
;
Variable
T    temperature [K]
x(i) liquid molar fraction
y(i) vapor molar fraction
;
equation
eq1,eq2 phase equilibrium equations
eq3,eq4 sumatory of component molar fractions
;
eq1.. f1l(T,P,x('water'),x('ethanol')) =e= f1v(T,P,y('water'),y('ethanol'));
eq2.. f2l(T,P,x('water'),x('ethanol')) =e= f2v(T,P,y('water'),y('ethanol'));
eq3.. sum(i,y(i)) =e= 1;
eq4.. sum(i,x(i)) =e= 1;

y.lo(i)=0; y.up(i)=1;
x.lo(i)=0; x.up(i)=1;

y.l(i)=0.5;
x.l(i)=0.5;
T.l=350;

model azeotrope /all/;
solve azeotrope using nlp minimizing T;
```

The obtained results are compared (Table 3) with experimental data taken from Tochigi et al. (1985). Despite the assumptions made to derive the model, for instance, the dependence of the NRTL interaction parameters with the temperature is neglected, a good agreement between the predicted and experimental results is observed.

Table 3. Comparison of the model-based results and experimental value (1.0132 bar)

	GAMS model	Experimental data [(*)]
Temperature [K]	351.5302	351.34
Composition (ethanol molar fraction)	0.8681	0.894

[(*)] Tochigi, K., Inoue, H., and Kojima, K. (1985). Determination of azeotropes in binary systems at reduced pressures. Fluid Phase Equilibria 22, 343–352.

Appendix 1. Supported compounds with the corresponding IDs.

ID	Name	ID	Name	ID	Name
1	Methane	505	O-xylene	1319	Isopropyl acetate
2	Ethane	506	M-xylene	1321	Vinyl acetate
3	Propane	507	P-xylene	1322	Methyl propionate
4	Isobutane	509	N-propylbenzene	1351	Methyl methacrylate
5	N-butane	510	Cumene	1357	N-pentyl acetate
7	N-pentane	511	O-ethyltoluene	1363	N-hexyl acetate
8	Isopentane	512	M-ethyltoluene	1366	Ethylene carbonate
9	Neopentane	513	P-ethyltoluene	1381	Dimethyl terephthalate
11	N-hexane	514	1,2,3-trimethylbenzene	1401	Dimethyl ether
12	2-methylpentane	515	1,2,4-trimethylbenzene	1402	Diethyl ether
13	3-methylpentane	516	Mesitylene	1403	Diisopropyl ether
14	2,2-dimethylbutane	518	N-butylbenzene	1404	Di-n-butyl ether
15	2,3-dimethylbutane	519	Isobutylbenzene	1405	Methyl tert-butyl ether
17	N-heptane	520	Sec-butylbenzene	1406	Di-sec-butyl ether
18	2-methylhexane	521	Tert-butylbenzene	1407	Methyl ethyl ether
19	3-methylhexane	522	O-cymene	1408	Methyl n-propyl ether
20	3-ethylpentane	523	M-cymene	1409	Isopropyl butyl ether
21	2,2-dimethylpentane	524	P-cymene	1410	Methyl isobutyl ether
22	2,3-dimethylpentane	525	O-diethylbenzene	1411	Methyl isopropyl ether
23	2,4-dimethylpentane	526	M-diethylbenzene	1421	1,4-dioxane
24	3,3-dimethylpentane	527	P-diethylbenzene	1427	Methyl tert-pentyl ether
25	2,2,3-trimethylbutane	530	1,2,3,4-tetramethylbenzene	1428	Tert-butyl ethyl ether
27	N-octane	531	1,2,3,5-tetramethylbenzene	1430	Ethyl tert-pentyl ether
28	2-methylheptane	532	1,2,4,5-tetramethylbenzene	1431	Methylal
29	3-methylheptane	544	P-diisopropylbenzene	1441	Ethylene oxide
30	4-methylheptane	558	Biphenyl	1442	1,2-propylene oxide
31	3-ethylhexane	576	2-ethyl-m-xylene	1447	Butyl vinyl ether
32	2,2-dimethylhexane	577	2-ethyl-p-xylene	1461	Anisole
33	2,3-dimethylhexane	578	4-ethyl-m-xylene	1472	Cumene hydroperoxide
34	2,4-dimethylhexane	579	4-ethyl-o-xylene	1479	Tetrahydrofuran
35	2,5-dimethylhexane	586	1-methyl-3-n-propylbenzene	1501	Carbon tetrachloride
36	3,3-dimethylhexane	587	1-methyl-4-n-propylbenzene	1502	Methyl chloride
37	3,4-dimethylhexane	601	Styrene	1503	Ethyl chloride
38	2-methyl-3-ethylpentane	701	Naphthalene	1504	Vinyl chloride

39	3-methyl-3-ethylpentane	702	1-methylnaphthalene	1521	Chloroform
40	2,2,3-trimethylpentane	703	2-methylnaphthalene	1522	1,1-dichloroethane
41	2,2,4-trimethylpentane	710	1-phenylnaphthalene	1523	1,2-dichloroethane
42	2,3,3-trimethylpentane	717	Fluoranthene	1524	1,1,2-trichloroethane
43	2,3,4-trimethylpentane	723	1-methylindene	1541	Trichloroethylene
44	2,2,3,3-tetramethylbutane	724	2-methylindene	1571	Monochlorobenzene
46	N-nonane	738	Fluorene	1572	O-dichlorobenzene
47	2,2,5-trimethylhexane	803	Indene	1573	M-dichlorobenzene
48	3,3,5-trimethylheptane	805	Phenanthrene	1574	P-dichlorobenzene
49	2,4,4-trimethylhexane	806	Chrysene	1592	1,2,4-trichlorobenzene
50	3,3-diethylpentane	807	Pyrene	1680	Bromobenzene
51	2,2,3,3-tetramethylpentane	808	Acenaphthene	1681	Methyl iodide
52	2,2,3,4-tetramethylpentane	820	Indane	1691	Iodobenzene
53	2,2,4,4-tetramethylpentane	899	Nitrous oxide	1701	Methylamine
54	2,3,3,4-tetramethylpentane	900	Nitrogen dioxide	1703	Trimethylamine
55	Squalane	901	Oxygen	1704	Ethylamine
56	N-decane	902	Hydrogen	1706	Triethylamine
62	Tert-butylcyclohexane	904	Nitrogen trioxide	1710	Diethylamine
63	N-undecane	905	Nitrogen	1722	Methyl DiEthanolAmine
64	N-dodecane	906	Nitrogen tetroxide	1723	Monoethanolamine
65	N-tridecane	908	Carbon monoxide	1724	Diethanolamine
66	N-tetradecane	909	Carbon dioxide	1725	Triethanolamine
67	N-pentadecane	910	Sulfur dioxide	1741	Ethylenediamine
68	N-hexadecane	911	Sulfur trioxide	1743	Diisopropylamine
69	N-heptadecane	912	Nitric oxide	1750	N-aminoethyl piperazine
70	N-octadecane	913	Helium-4	1760	Nitromethane
71	N-nonadecane	914	Argon	1761	Nitroethane
72	2,2-dimethyloctane	915	Air	1762	1-nitropropane
73	N-eicosane	917	Fluorine	1763	2-nitropropane
74	N-heneicosane	918	Chlorine	1769	1-nitrobutane
75	N-docosane	919	Neon	1771	Hydrogen cyanide
76	N-tricosane	920	Krypton	1772	Acetonitrile
77	N-tetracosane	922	Bromine	1773	Propionitrile
78	N-pentacosane	924	Ozone	1774	Acrylonitrile
79	N-hexacosane	959	Xenon	1775	Methacrylonitrile
80	N-heptacosane	1001	Formaldehyde	1778	O-nitrotoluene
81	N-octacosane	1002	Acetaldehyde	1779	P-nitrotoluene
82	N-nonacosane	1003	Propanal	1780	M-nitrotoluene
85	3-methylnonane	1005	Butanal	1791	Pyridine
86	2-methylnonane	1006	2-methylpropanal	1792	Aniline
87	4-methylnonane	1007	Pentanal	1801	Methyl mercaptan
88	5-methylnonane	1008	Heptanal	1802	Ethyl mercaptan
91	2-methyloctane	1009	Hexanal	1803	N-propyl mercaptan
92	3-methyloctane	1051	Acetone	1804	Tert-butyl mercaptan

93	4-methyloctane	1052	Methyl ethyl ketone	1805	Isobutyl mercaptan
94	3-ethylheptane	1053	3-pentanone	1806	Sec-butyl mercaptan
96	2,2-dimethylheptane	1054	Methyl isobutyl ketone	1807	N-hexyl mercaptan
102	Cyclobutane	1057	3-heptanone	1810	Isopropyl mercaptan
104	Cyclopentane	1058	4-heptanone	1813	Methyl ethyl sulfide
105	Methylcyclopentane	1059	3-hexanone	1814	Methyl n-propyl sulfide
107	Ethylcyclopentane	1060	2-pentanone	1815	Methyl t-butyl sulfide
108	1,1-dimethylcyclopentane	1061	Methyl isopropyl ketone	1816	Methyl t-pentyl sulfide
109	Cis-1,2-dimethylcyclopentane	1062	2-hexanone	1817	Di-n-propyl sulfide
110	Trans-1,2-dimethylcyclopentane	1063	2-heptanone	1818	Diethyl sulfide
111	Cis-1,3-dimethylcyclopentane	1064	5-methyl-2-hexanone	1820	Dimethyl sulfide
112	Trans-1,3-dimethylcyclopentane	1066	3,3-dimethyl-2-butanone	1821	Thiophene
114	N-propylcyclopentane	1068	Diisobutyl ketone	1824	Diethyl disulfide
115	Isopropylcyclopentane	1069	Diisopropyl ketone	1828	Dimethyl disulfide
116	1-methyl-1-ethylcyclopentane	1080	Cyclohexanone	1829	Di-n-propyl disulfide
122	N-butylcyclopentane	1100	Ketene	1844	Dimethyl sulfoxide
137	Cyclohexane	1101	Methanol	1845	Sulfolane
138	Methylcyclohexane	1102	Ethanol	1851	Acetyl chloride
140	Ethylcyclohexane	1103	1-propanol	1854	Dichloroacetyl chloride
141	1,1-dimethylcyclohexane	1104	Isopropanol	1855	Trichloroacetyl chloride
142	Cis-1,2-dimethylcyclohexane	1105	1-butanol	1876	N,n-dimethylformamide
143	Trans-1,2-dimethylcyclohexane	1106	2-methyl-1-propanol	1886	Nitrobenzene
144	Cis-1,3-dimethylcyclohexane	1107	2-butanol	1889	Furfural
145	Trans-1,3-dimethylcyclohexane	1108	2-methyl-2-propanol	1893	Carbonyl sulfide
146	Cis-1,4-dimethylcyclohexane	1109	1-pentanol	1894	Phosgene
147	Trans-1,4-dimethylcyclohexane	1110	2-pentanol	1903	Nitric acid
149	N-propylcyclohexane	1111	2-methyl-2-butanol	1904	Hydrogen chloride
152	N-butylcyclohexane	1112	2-methyl-1-butanol	1907	Hydrogen iodide
153	Cis-decahydronaphthalene	1113	2,2-dimethyl-1-propanol	1911	Ammonia
154	Trans-decahydronaphthalene	1114	1-hexanol	1921	Water
201	Ethylene	1125	1-heptanol	1922	Hydrogen sulfide
202	Propylene	1151	Cyclohexanol	1938	Carbon disulfide
204	1-butene	1181	Phenol	1940	Sulfur hexafluoride
205	Cis-2-butene	1182	O-cresol	2252	2-methyl-1-heptene
206	Trans-2-butene	1183	M-cresol	2367	Propylene carbonate
207	Isobutene	1184	P-cresol	2391	Dimethyl carbonate
209	1-pentene	1201	Ethylene glycol	2717	Diethylenetriamine
210	Cis-2-pentene	1202	Diethylene glycol	2732	N-aminoethyl ethanolamine
211	Trans-2-pentene	1203	Triethylene glycol	2743	2,4-dinitrotoluene
212	2-methyl-1-butene	1204	Tetraethylene glycol	2744	2,6-dinitrotoluene
213	3-methyl-1-butene	1231	Glycerol	2745	3,4-dinitrotoluene
214	2-methyl-2-butene	1241	1,4-butanediol	2747	2,4,6-trinitrotoluene
216	1-hexene	1252	Acetic acid	2748	2,5-dinitrotoluene
217	Cis-2-hexene	1253	Propionic acid	2749	3,5-dinitrotoluene

218	Trans-2-hexene	1255	Oxalic acid	2750	P-phenylenediamine
221	2-methyl-1-pentene	1256	N-butyric acid	2752	Piperazine
227	4-methyl-cis-2-pentene	1277	Acrylic acid	2856	N,n-dimethylacetamide
228	4-methyl-trans-2-pentene	1278	Methacrylic acid	3801	Di-tert-butyl disulfide
234	1-heptene	1281	Benzoic acid	3813	Ethyl methyl disulfide
250	1-octene	1282	O-toluic acid	3814	Ethyl propyl disulfide
259	1-nonene	1283	P-toluic acid	3819	Diphenyl disulfide
261	1-undecene	1284	Salicylic acid	4865	Trichloroacetaldehyde
270	Cyclohexene	1285	Adipic acid	4868	Dichloroacetaldehyde
301	Propadiene	1286	Maleic acid	6861	Diethylethanolamine
302	1,2-butadiene	1287	Phthalic acid	6862	Methylethanolamine
303	1,3-butadiene	1289	Terephthalic acid	6863	Dimethylethanolamine
309	Isoprene	1291	Acetic anhydride	6864	Diisopropanolamine
316	Dicyclopentadiene	1298	Maleic anhydride	13125	DiPhenyl Carbonate
401	Acetylene	1301	Methyl formate	20101	2-Methyl-2-Heptanol
402	Methylacetylene	1302	Ethyl formate	22158	2-Methoxy-2-Methyl-Heptane
403	Ethylacetylene	1303	N-propyl formate	22587	Ethyl Phenyl Carbonate
404	Dimethylacetylene	1312	Methyl acetate	23498	Methyl Ethyl Carbonate
418	Vinylacetylene	1313	Ethyl acetate	27991	Methyl Phenyl Carbonate
501	Benzene	1314	N-propyl acetate	28366	DiEthyl Carbonate
502	Toluene	1315	N-butyl acetate		
504	Ethylbenzene	1316	Isobutyl acetate		

Appendix 2. Library definition for the presented case studies

Case study 1 involves a mixture of water and ethanol. As shown in Appendix 1, the compounds IDs are 1921 and 1102 respectively.

```
$onecho > NRTLidealID.txt
ID1 1921
ID2 1102
$offecho
$onecho > NRTLidealaij.txt
a12 1241.7396
a21 -57.9601
$offecho
$onecho > NRTLidealalphaij.txt
alpha12 0.2937
$offecho
$funclibin NRTL NRTLideal.dll
function h_liq   /NRTL.h_liq  /;
function f1_liq  /NRTL.f1_liq /;
function f2_liq  /NRTL.f2_liq /;
function h_vap   /NRTL.h_vap  /;
function rho_vap /NRTL.rho_vap/;
function f1_vap  /NRTL.f1_vap /;
function f2_vap  /NRTL.f2_vap /;
```

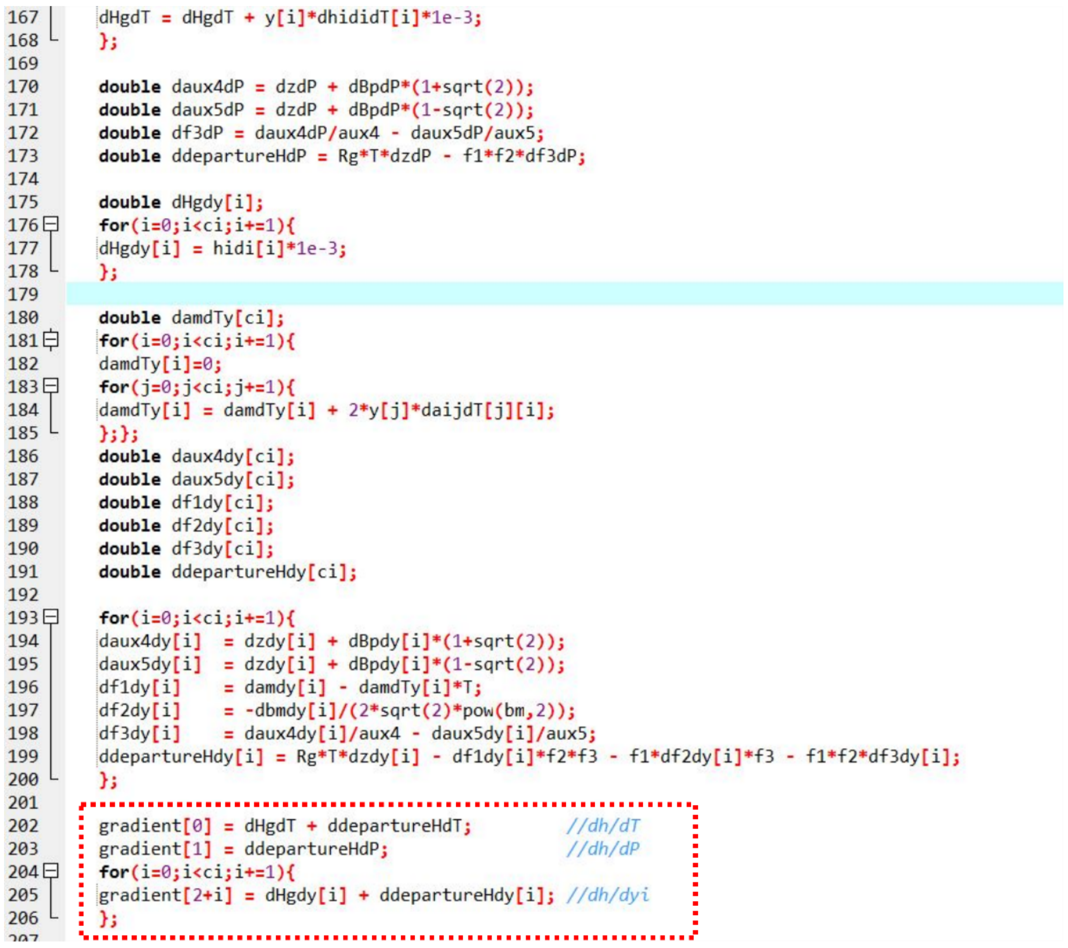
In case study 2, a mixture of n-pentane, n-hexane and n-heptane is analyzed and the compounds IDs are 7, 11 and 17 respectively.

```
$onecho > PengRobinsonID.txt
ID1 7
ID2 11
ID3 17
$offecho
$onecho > PengRobinsonaij.txt
a12 0.000393
a13 0.001373
a23 0.000297
$offecho
$funclibin PengRobinson PengRobinson.dll
function h_liq   /PengRobinson.h_liq  /;
function f1_liq  /PengRobinson.f1_liq /;
function f2_liq  /PengRobinson.f2_liq /;
function f3_liq  /PengRobinson.f3_liq /;
function h_vap   /PengRobinson.h_vap  /;
function rho_vap /PengRobinson.rho_vap/;
function f1_vap  /PengRobinson.f1_vap /;
function f2_vap  /PengRobinson.f2_vap /;
function f3_vap  /PengRobinson.f3_vap /;
```

Finally, in the comparison example (Section 3.5), a mixture of methanol and ethanol is used. According to Appendix 1, the compounds IDs are 1101 and 1102 respectively.

```
$onecho > NRTLidealID.txt
ID1 1101
ID2 1102
$offecho
$onecho > NRTLidealaij.txt
a12 -327.9991
a21 376.2667
$offecho
$onecho > NRTLidealalphaij.txt
alpha12 0.3057
$offecho
$funclibin NRTL NRTLideal.dll
function f1_liq  /NRTL.f1_liq /;
function f2_liq  /NRTL.f2_liq /;
function f1_vap  /NRTL.f1_vap /;
function f2_vap  /NRTL.f2_vap /;
function h_liq   /NRTL.h_liq /;
function h_vap   /NRTL.h_vap /;
```

As mentioned in the manuscript, the gradient vector and the Hessian matrix were implemented analytically in each extrinsic function in the C programming language. As illustration, extracts of the source codes for computing the gradient vector and Hessian matrix in the extrinsic function corresponding to the liquid enthalpy in the Peng-Robinson library (*PengRobinson.dll*) are presented in Figs. S1 and S2, respectively.

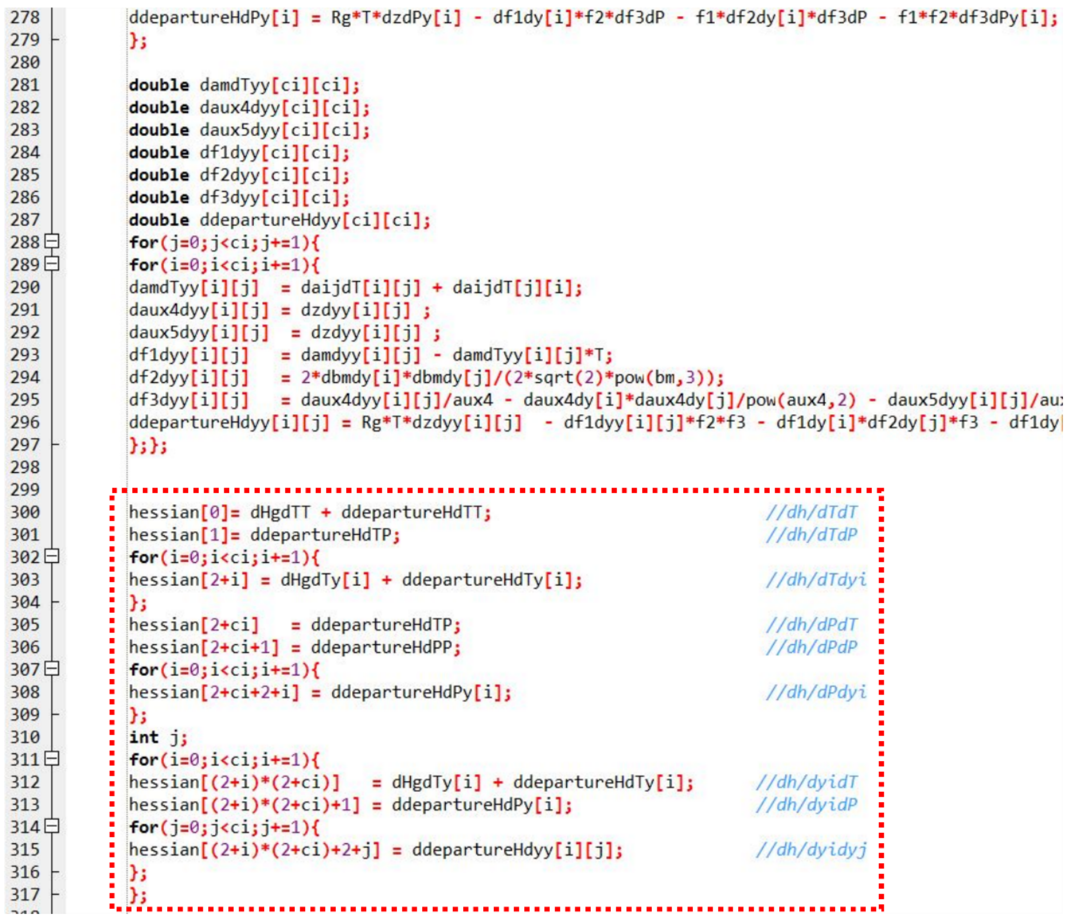


```
dHgdT = dHgdT + y[i]*dhididT[i]*1e-3;
};

double daux4dP = dzdP + dBpdP*(1+sqrt(2));
double daux5dP = dzdP + dBpdP*(1-sqrt(2));
double df3dP = daux4dP/aux4 - daux5dP/aux5;
double ddepartureHdP = Rg*T*dzdP - f1*f2*df3dP;

double dHgdy[i];
for(i=0;i<ci;i+=1){
dHgdy[i] = hidi[i]*1e-3;
};

double damdTy[ci];
for(i=0;i<ci;i+=1){
damdTy[i]=0;
for(j=0;j<ci;j+=1){
damdTy[i] = damdTy[i] + 2*y[j]*daijdT[j][i];
};};
double daux4dy[ci];
double daux5dy[ci];
double df1dy[ci];
double df2dy[ci];
double df3dy[ci];
double ddepartureHdy[ci];

for(i=0;i<ci;i+=1){
daux4dy[i]  = dzdy[i] + dBpdy[i]*(1+sqrt(2));
daux5dy[i]  = dzdy[i] + dBpdy[i]*(1-sqrt(2));
df1dy[i]    = damdy[i] - damdTy[i]*T;
df2dy[i]    = -dbmdy[i]/(2*sqrt(2)*pow(bm,2));
df3dy[i]    = daux4dy[i]/aux4 - daux5dy[i]/aux5;
ddepartureHdy[i] = Rg*T*dzdy[i] - df1dy[i]*f2*f3 - f1*df2dy[i]*f3 - f1*f2*df3dy[i];
};

gradient[0] = dHgdT + ddepartureHdT;          //dh/dT
gradient[1] = ddepartureHdP;                  //dh/dP
for(i=0;i<ci;i+=1){
gradient[2+i] = dHgdy[i] + ddepartureHdy[i]; //dh/dyi
};
```

Figure S1. Extract of the source code for computing the gradient vector corresponding to the liquid enthalpy in the Peng-Robinson library

In Fig. S1, the dotted box indicates how each position of the gradient vector of the enthalpy is calculated. The first position (gradient [0]) corresponds to the function derivative with respect to the first argument (temperature), the second position (gradient [1]) refers to the function derivative with respect to the second argument (pressure), and the third and successive positions refer to the function derivatives with respect to the third and successive arguments (concentration of each component).

Analogously, Fig. S2 shows the piece of source code corresponding to the computation of the Hessian matrix for the same example.



```
ddepartureHdPy[i] = Rg*T*dzdPy[i] - df1dy[i]*f2*df3dP - f1*df2dy[i]*df3dP - f1*f2*df3dPy[i];
};

double damdTyy[ci][ci];
double daux4dyy[ci][ci];
double daux5dyy[ci][ci];
double df1dyy[ci][ci];
double df2dyy[ci][ci];
double df3dyy[ci][ci];
double ddepartureHdyy[ci][ci];
for(j=0;j<ci;j+=1){
for(i=0;i<ci;i+=1){
damdTyy[i][j]  = daijdT[i][j] + daijdT[j][i];
daux4dyy[i][j] = dzdyy[i][j] ;
daux5dyy[i][j]  = dzdyy[i][j] ;
df1dyy[i][j]   = damdyy[i][j] - damdTyy[i][j]*T;
df2dyy[i][j]   = 2*dbmdy[i]*dbmdy[j]/(2*sqrt(2)*pow(bm,3));
df3dyy[i][j]   = daux4dyy[i][j]/aux4 - daux4dy[i]*daux4dy[j]/pow(aux4,2) - daux5dyy[i][j]/au
ddepartureHdyy[i][j] = Rg*T*dzdyy[i][j]  - df1dyy[i][j]*f2*f3 - df1dy[i]*df2dy[j]*f3 - df1dy
};};


hessian[0]= dHgdTT + ddepartureHdTT;                          //dh/dTdT
hessian[1]= ddepartureHdTP;                                   //dh/dTdP
for(i=0;i<ci;i+=1){
hessian[2+i] = dHgdTy[i] + ddepartureHdTy[i];                 //dh/dTdyi
};
hessian[2+ci]    = ddepartureHdTP;                            //dh/dPdT
hessian[2+ci+1] = ddepartureHdPP;                             //dh/dPdP
for(i=0;i<ci;i+=1){
hessian[2+ci+2+i] = ddepartureHdPy[i];                        //dh/dPdyi
};
int j;
for(i=0;i<ci;i+=1){
hessian[(2+i)*(2+ci)]    = dHgdTy[i] + ddepartureHdTy[i];    //dh/dyidT
hessian[(2+i)*(2+ci)+1] = ddepartureHdPy[i];                 //dh/dyidP
for(j=0;j<ci;j+=1){
hessian[(2+i)*(2+ci)+2+j] = ddepartureHdyy[i][j];            //dh/dyidyj
};
};
```

Figure S2. Extract of the source code for computing the Hessian matrix corresponding to the liquid enthalpy in the Peng-Robinson library.