Proceedings of the XXII Congreso Interamericano de Ingeniería Química y V Congreso Argentino de Ingeniería Quimica (XXII CIIQ y V CAIQ), Buenos Aires, Argentina (2006).

A NEW TOOL FOR OPTIMIZATION OF CHEMICAL PROCESSES USING RIGOROUS MODELS

E.C. do Valle, R.P. Soares, T.F. Finkler, A.R. Secchi*

Group of Integration, Modeling, Simulation, Control and Optimization of Processes – GIMSCOP

Department of Chemical Engineering,

Federal University of Rio Grande do Sul - UFRGS

Rua Luiz Englert, s/n CEP: 90040-040 - Porto Alegre - RS - BRAZIL

E-mail: arge@enq.ufrgs.br

Abstract. In the last decades, process simulators software have become a very useful tool which allows process engineers to evaluate projects practical operability, potential increases in production, potential reduction in the operational costs and in residues generation. Although the use of simulation and optimization tools together may improve even more the functionality of both tools, it is well known that the employment of optimization techniques in these software is limited by the large number of variables that compose the models, since a typical industrial rigorous process model can have thousands of variables (from 1.000 to 100.000). Considering these and others difficulties, the aim of this work is to integrate optimization routines into a process simulator to solve complexes chemical and petrochemical process optimization problems in a fast and robust way. The software EMSO (Environment for Modeling, Simulation and Optimization), is an environment for development and evaluation of dynamic and stationary models. This software provides several

functionalities that evaluate a DAE (Differential-Algebraic Equations) system in a fast and efficient way. Trying to take advantage of EMSO's efficiency to solve complexes DAE systems, NLP solvers were interfaced with the software in order to minimize the computational effort required to achieve the solution of optimization problems. The efficiency of the proposed methodology was tested with two case studies. With the optimization solvers incorporated to EMSO, the software becomes a complete tool for chemical and petrochemical processes modeling, simulation, and optimization.

Keywords: Modeling, Simulation, Optimization.

1. Introduction

Nowadays, chemical process simulator is an important tool for the evaluation of operational scenarios in the industry. Regarding industrial applications, process simulators can be used in synthesis, operation, and optimization (Ondrey, 2005). In this context, the use of process simulators presents some advantages such as resources economy, pollution prevention, and process security improvement.

The use of simulation tools in the project stage is especially important because in this phase the potential of energy (cold and hot utilities) and resources (water and raw material) reductions are higher (Hertz, 1995). However, in the project stage process information is reduced, as can be seen in Figure 1. For that reason, the use of simulation tools in the synthesis stage is important to evaluate the process behavior in different scenarios even before the plant start-up.

* To whom all correspondence should be addressed

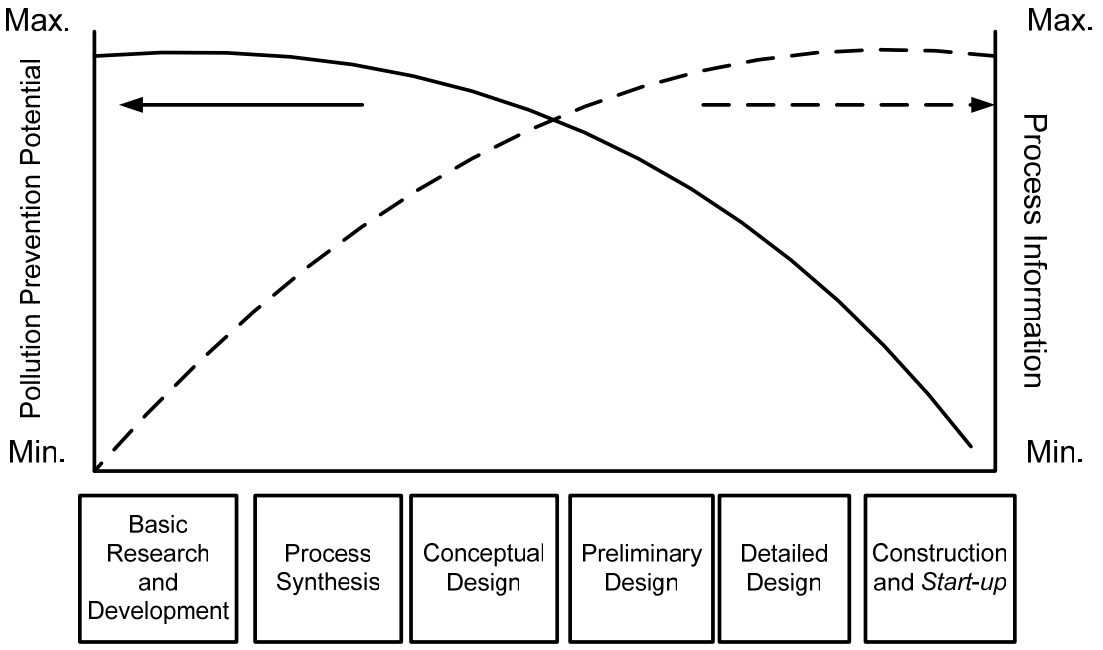

Figure 1: Process information and the energy and pollutants reduction potential in several project stages (Hertz, 1995).

During the plant operation it is possible, through the use of simulation tools, to evaluate the process behavior in several scenarios, such as:

•Unit start-up and shutdown.

•Grades transition.

•Operation in the production capacity or security limits.

•Raw-material, equipment or configuration modifications.

In the optimization stage, it is possible, through the use of process simulators, find the best operational condition to attain some production, environment, security or economic goal and which actions must be performed to reach such objective.

The use of simulation tools has been largely used in the Academia for scientific purposes. In 2005 were published 222 articles about modeling and simulation by the American Chemical Society Press in the Industrial and Engineering Chemistry Research Journal. The use of such tools in the engineering teaching is also important in order to demonstrate to the students some important concepts, as presented in Nance and Balci (2001), Banks (2001), Vin and Jägstam (2001), Yang (2001), Nance (2000).

Several process simulators are available commercially; many of them have closed models with only some parameters to be set by the user. This class of simulators, called block oriented, some times do not present desired results, since, due to complexity, the process cannot be well represented by the closed model. Examples of these complex equipments are: polymerization, heterogeneous and membrane reactors, reactive distillation columns, bioreactors, and others. In the case where the process cannot be well represented by a model, some applications, such as optimization do not lead to desired results or cannot be performed. To avoid this problem, the alternative is to use equation-based simulators. With these simulators the user can model the process in a more detailed way (since one can write the own equations), and therefore represent more complex systems. With these tools, a good knowledge of the physical and chemical phenomena involved in the process is needed and also robust solving mathematical methods are necessary to solve the problem. A review of both class of simulation software and commercial available tools are presented in Tu and Rinard (2006). Since the detailed modeling is expected to become the major bottleneck in the widespread use of model-based techniques in industrial practice (Tu and Rinard, 2006), it is expected that the equation-based simulators can, someway, simplify this task.

In this context the software EMSO, Environment for Modeling, Simulation and Optimization, (Soares and Secchi, 2003) is a generic steady-state and dynamic simulator, whose main application is the modeling of chemical and petrochemical process modeling. The simulator is object oriented and the code can be reused between flowsheets, simplifying the modeling process. For example: a model of a mixer that considers the energy balance of the mixture can be modeled reusing a model of a mixer that considers only mass and components balance with the energy balance added. Since EMSO has a dynamic simulator feature, it is possible to monitor the process variables along the time, allowing the simulation of plant start-up and shutdown and also operating-point transition. The software has many features such as its own graphical interface which can plot graphical results, a differential-algebraic equation systems characterization, the use of sparse algebra algorithms, automatic differentiation, detection and solving high-index differential-algebraic equation system, a unit

consistency checker, a plug-in system which allows the use of external routines (such as solvers and thermodynamic packages) and others. Nowadays, EMSO has a vast rigorous model library, which includes the main equipments used in the chemical and petrochemical industry such as separators (flash, distillation columns), heat exchangers, reactors, pumps, valves, controllers, and many others. The models presented in the library are opened and can be viewed edited and added by the users. A simulation result of the start-up of a distillation column with 2 components and 8 stages is presented in Figure 2.

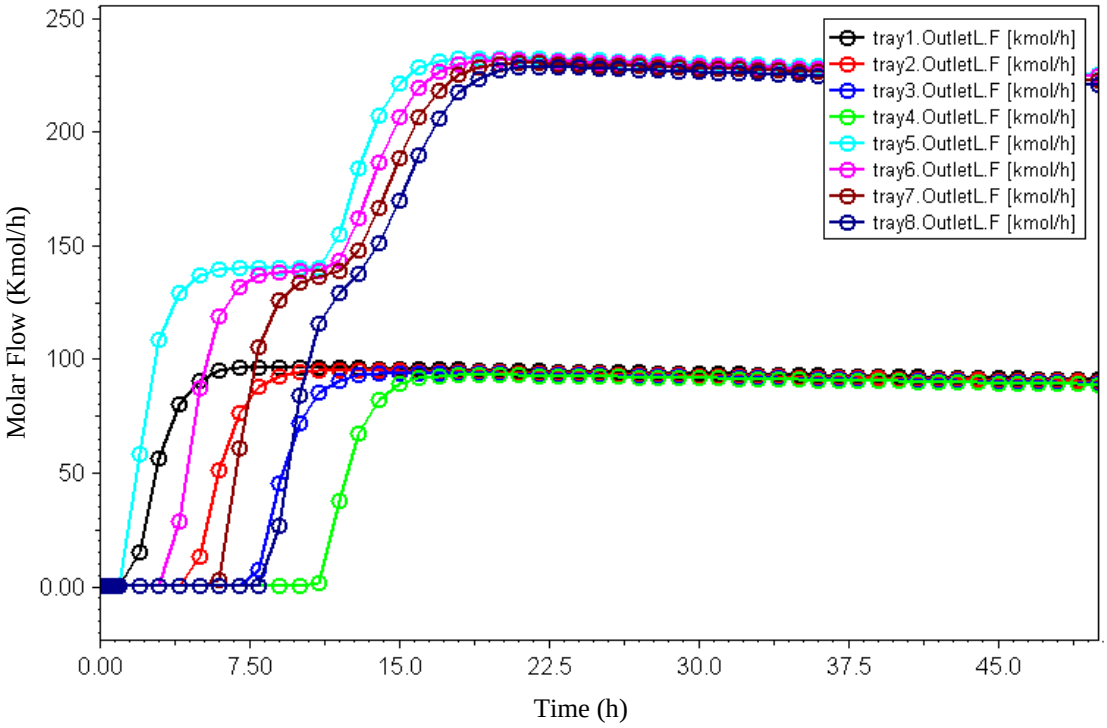

Figure 2: Simulation result of a 2 components (isobutane and n-pentane) and 8 stages distillation column start-up.

Integrating process simulators with external optimization routines increases the software features, being possible to:

•Perform steady-state optimization that can determine the optimal operating condition to reach some desired goals.

•Perform parameter estimation (kinetic constants, efficiency, and others) based on process data fed into the simulator.

•Perform process data reconciliation (such as: flowrates, composition, temperatures).

•Perform process synthesis (such as: heat exchanger, mass exchange networks, reaction/separation synthesis, distillation columns sequencing and others) based on optimization methods that can handle continuous and discrete variables.

•Perform dynamic optimization which, in the chemical processes case, can help to determine which of the process variables must be modified and in which sequence in time, to reach a desired goal.

The successes of modern optimization strategies in process engineering, constantly motivates the desire to formulate and solve large problems (Grossmann and Biegler, 2004). However, many difficulties appear when optimization is performed using rigorous models. One of them is due to computational constraints: rigorous models requires the solution of large-scale problem (typically between 1.000 to 100.000 variables) and can present high sparsity. The presence of non-linearity in the rigorous models also introduces a difficulty in the problem solving. To avoid these problems, it is common to use rigorous models to simulation and simplified ones to optimization, which may not represent accurately the process.

In this context, the aim of this work is to present the implementation of an integration between EMSO and IPOPT (Wächter, 2002), a large scale optimization routine.

2. Methods

The communication between the optimization solver and EMSO was performed in three stages:

•Development of the optimization language.

•Choosing the optimization solver.

•Development and implementation of a communication interface between the optimization solver and the simulator.

It was defined that the optimization problem could be written in two ways: in the first one, the user defines and writes all the variables and equations involved in the optimization problem, whereas, in the second, the user imports the variables and equations of an already defined simulation/flowsheet (at this moment, only steady-state flowsheets can be imported). In this last case, it is necessary to free some variables in order to have the necessary degrees or freedom to run the optimization.

2.1. Optimization Language

The optimization problem starts with the keyword “Optimization” and ends with the word “end”. The problem description is divided in five subsections:

- VARIABLES: where the decision, or free, variables that integrate the problem are declared. Here it is also defined the upper and lower bounds and also the initial guesses for the variables.
- MINIMIZE or MAXIMIZE: where the objective function is written. It is possible to perform a maximization or minimization of multi-objective functions.
- EQUATIONS: where the equality and inequality constraints are written.
- OPTIONS: where the solver specific options are set.
- FREE: in the case where the equations of a flowsheet are used (such as mass and energy balances) it is possible define here which of the variables are allowed to be free by the optimization. In the case where all equations are written in the EQUATIONS section, and all the variables are declared in the VARIABLES section, it is not necessary to use this section.

In order to present the language, the optimization test problem proposed by Hock and Schittkowski (1981), listed in Eq. (1) is listed below.

$$
\begin{aligned}
&\min_{x_1,x_2,x_3,x_4} x_1.x_4(x_1 + x_2 + x_3) + x_3 \\
&s.a. \\
&1 < x_i < 5, i = 1..4 \\
&x_1.x_2.x_3.x_4 \geq 25 \\
&x_1^2 + x_2^2 + x_3^2 + x_4^2 = 40
\end{aligned}
\tag{1}
$$

The problem above, written in the proposed language have the following form:

```
Optimization hs71
    VARIABLES
    x1 as Real(Default=1, Lower=1, Upper=5);
    x2 as Real(Default=5, Lower=1, Upper=5);
    x3 as Real(Default=5, Lower=1, Upper=5);
    x4 as Real(Default=1, Lower=1, Upper=5);

    MINIMIZE
    x1*x4*(x1+x2+x3) + x3;

    EQUATIONS
    x1*x2*x3*x4 > 25;

    x1*x1 + x2*x2 + x3*x3 + x4*x4 = 40;

    OPTIONS
    NLPSolver = "opt_emso";
    outputLevel = "all";
    relativeAccuracy = 1e-6;
end
```

In the case where previously built flowsheets are used (in the case below there are two disconnected units, where “unit1” has a distillation column, and “unit2” has a reactor, connected by Problem_1) the problem has the following form:

```
Optimization Problem_1
  FLOWSHEETS
   F1 as unit1;
   F2 as unit2;

  MINIMIZE
   Fixed_Cost + Var_Cost;

  MAXIMIZE
   Selectivity;

  EQUATIONS
   Fixed_Cost = F1.FC + F2.FC;
   Var_Cost = F1.VC + F2.VC;
   Selectivity = F2.REACTOR.S;
   F2.MIXER.Fin1 = F1.COLUMN.Ftop;
   340 < F2.REACTOR.T(End) < 410;

  FREE
   F1.COLUMN.Fin;
   F2.REACTOR.V;
End
```

2.2. Optimization Solver

The use of object-oriented software libraries for optimization is a future tendency, as presented in Grossmann and Biegler (2004), since one of the goals of object-oriented software is to allow plug and play features in the formulation of the optimization models. In this context, two solvers were evaluated: OPT++ (Meza and Oliva, to appear) and IPOPT (Wächter, 2002). OPT++ is a free software (LGPL license), written in C++, developed by Sandia National Labs. IPOPT is developed as part of the COIN-OR project sponsored by IBM, it is also free (CPL license, less restricted when it comes to distribution) and also written in C++. Both are routines that handle with non-linear equality and inequality constraints (very common in Chemical Engineering problems), and use interior point algorithms (see Rao, 1996). The main advantage of IPOPT over OPT++ is that the former is specially designed to deal with large-scale problem, also common in Chemical Engineering problems when an equipment model is written in a rigorous way. As the simulator EMSO, IPOPT has special sparse algebra routines that reduce the memory used to allocate the Jacobian matrix, so this data is passed from the simulator to the optimizator in a more efficient way.

2.3. Communication Interface between the Simulator and the Optimization Solver

The communication interface was made to use the automatic differentiation (for the objective function and constraints gradients) and the sparse algebra routines of EMSO. The optimization problem solutions are obtained by the following steps:

1. Problem initialization by the optimization solver (OS).
2. An automatic differentiation of the objective function and constraints is performed by EMSO and the structure of the gradients/Jacobian matrix is passed to the OS.
3. The OS requests the objective function and the constraints (equality and inequality) residuals and gradients (Jacobian) to EMSO.

4. EMSO calculates the objective function and the constraints residuals and gradients and passes the values to the OS as sparse (compacted) matrix.

5. A new step or search direction is calculated by the OS. Steps 2 to 4 are repeated until convergence or non-convergence condition is met.

6. The OS returns the results to EMSO.

7. Results are published by EMSO.

In the case where a flowsheet optimization is executed, it is possible to solve the system of equations (resulting from the flowsheet) considering only the equality constraints (as a non-linear-algebraic system of equations, or NLA) before the optimization, in order to present a feasible initial guess for the OS. It is important to notice that in the case where thermodynamic properties are needed, EMSO make a call to the thermodynamic property package that retrieves (from a data bank) or calculates the desired property, which may take extra computing time.

Since EMSO does not provide the Hessian matrix by automatic differentiation, the OS must calculate it by finite differences and, therefore, it is possible to have loss in the Hessian matrix accuracy and an extra computational time is required comparing with a case where this matrix is already provided.

In the next section, some tests were carried out to evaluate the EMSO-OS integration performance.

3. Results and Discussion

The integration between EMSO and IPOPT was tested with two literature examples. The optimizations were performed in a Pentium 4 3.0 GHz HT with 512 MB of RAM memory using Windows XP as operational system. The thermodynamic property package used was VRTherm (VRTech, 2005).

3.1. A Simple Flash Example

The starting example implemented is a TP flash (where the temperature and pressure are specified). This example is based in the work of Gani (1986) where the components, some models and operating conditions were obtained from that work. The components in the feed stream are 1,3-butadiene, isobutene, n-pentane, 1-pentene, 1-hexene and benzene. For the thermodynamic calculations, Peng-Robinson equation of state was used for the liquid and vapour phases. The equipment is presented in Figure 3.

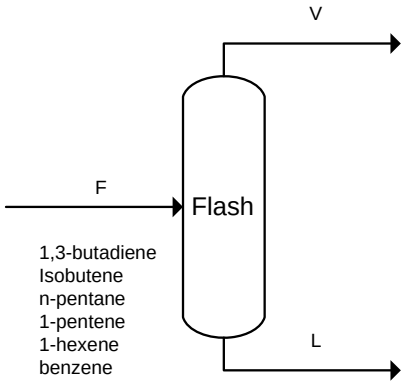

Figure 3: A flash separation example.

The flash model is represented by the following code:

```
  FlowSheet FlashSteadyTest

  PARAMETERS
      PP    as CalcObject(Brief = "Physical Properties",
File="vrpp");
      NComp as Integer;

  VARIABLES
      Q     as heat_rate (Brief="Heat supplied");
      light as fraction;

  SET
      PP.Components = ["1,3-butadiene", "isobutene", "n-pentane",
"1-pentene", "1-hexene", "benzene"];
      PP.LiquidModel = "PR";
      PP.VapourModel = "PR";
      NComp = PP.NumberOfComponents;
```

```
DEVICES
    fl as flash_Steady;
    s1 as stream_therm;

CONNECTIONS
    s1 to fl.Inlet;
    Q to fl.Q;

EQUATIONS
    light = fl.OutletV.z(1)+fl.OutletV.z(2);

SPECIFY
    s1.F = 496.3 * "kmol/h";
    s1.T = 338 * "K";
    s1.P = 507.1 * "kPa";
    s1.v = 0.1380;
    s1.z = [0.2379,0.3082,0.09959,0.1373,0.08872,0.1283];
    fl.OutletL.P = 2.5 * "atm";
    fl.OutletL.T = 315.6 * "K";

OPTIONS
    mode = "steady";
    outputLevel = "all";
end

Optimization FlashOpt1 as FlashSteadyTest
MAXIMIZE
    light;
    FREE
    fl.OutletL.T;
    EQUATIONS
    fl.OutletL.T < 315 * "K";
    fl.OutletL.T > 300 * "K";
    OPTIONS
    outputLevel = "high";
end
```

In this optimization problem, first it is defined a flowsheet, named “FlashSteadyTest”, and then an optimization problem that uses the equations of this previously defined flowsheet, named “Flashopt1”.

In the section “PARAMETERS”, the first line defines an object named “PP” that is a link to the thermodynamic property package VRTherm, which is included in the “vrpp” dynamic link library.

In the “VARIABLES” section, the variables of the flowsheet are defined.

In the “SET” section, components and the thermodynamic models for the liquid and vapour phases are defined.

In the “DEVICES” section, two objects are defined: a TP flash vessel (or a flash were temperature and pressure are specified), named “f1”, and a stream, named “s1”, representing the feed stream.

In the “EQUATIONS” section, an equation for “light” is defined as the sum of fraction of components 1 and 2. This definition will be used later in the optimization problem.

In the “SPECIFY” section, the specifications are defined in order to make the system of equations consistent (the degrees of freedom of the NLA system of equations must be zero).

The “OPTIONS” section defines the solver specific options, where for this example it is possible to perform a steady-state simulation.

This is a small system of non-linear algebraic equation problem with 36 equations and with the same number of variables. To write the optimization problem, an objective function, extra constraints and a decision (free) variable must be defined. The objective, in this case, is to maximize the 1,3-butadiene and isobutene purity in the vaporized stream and it is represented in the section “MAXIMIZE”, with the previously defined equation named “light”. Since the degree of freedom of the original problem is already zero, it is necessary to free, at least, one variable in order to solve the problem. The variable chosen is the liquid outlet temperature and it’s shown in the section “FREE”. It was also added two inequality constraints to the problem: the temperature must be between 300K and 310K, defined in the section “EQUATIONS”. In Table 1 and Table 2 the optimization performance and the optimization results are presented, respectively.

Table 1: Performance results of the optimization for the flash separation problem.

Run	Absolute Tolerance NLP only	Initial guess provided by NLA	Converged	Time (s) NLP only	Iterations	Function evaluations (objective and constraints)	Objective Jacobian evaluations	Constraints Jacobian Evaluations
1	1.0E-5	No	Yes	0.110	11	11	12	12
2	1.0E-5	Yes	Yes	0.156	9	12	11	11

Table 2: Results of the optimization for the flash separation problem.

Run	Light fraction in vapour stream	Temperature (K)
Initial guess	0.793	315.6
1	0.861	301.73
2	0.849	309.9

It is interesting to notice that, in this case, when the NLA solver provides a feasible initial guess, the optimization failed and did not converge to the global optimum (temperature = 301.73 K) and when the initial guess was not provided by the NLA solver (unfeasible initial guess), the optimization converged successfully to the global optimum.

Although it is a small problem, the Jacobian matrix of the equality constraints has 1296 elements (36 equations and 36 variables). The use of sparse algebra can significantly reduce the floating-point operations since only 180 elements of this matrix are nonzero, almost 7 times smaller than the full matrix.

3.2. The Ammonia Plant Example

This example uses the Ammonia production process presented by Biegler et al. (1997) using rigorous models. For the thermodynamic calculation, the Assimetric Peng-Robinson equation of state for the liquid and vapour phases was used. The problem has 134 variables and 134 equations and it is presented in Figure 4.

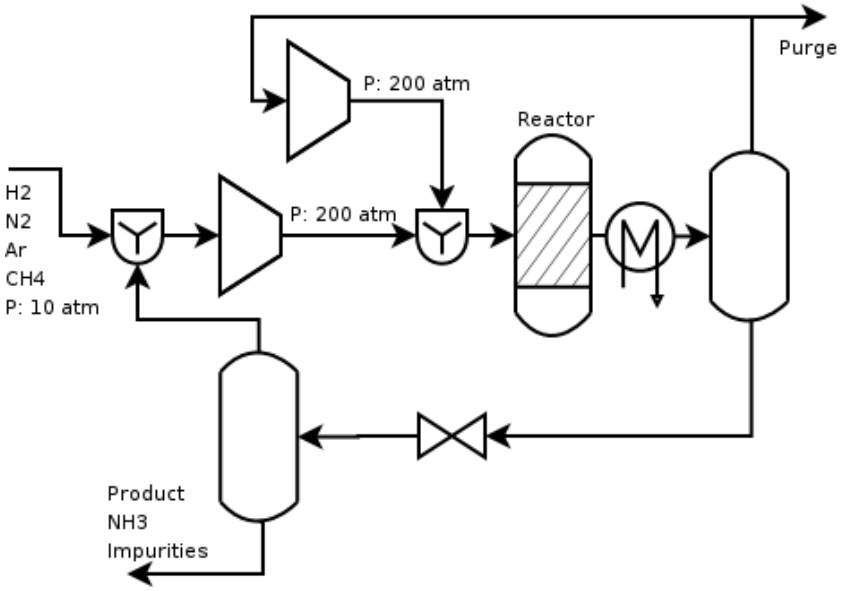

Figure 4: Ammonia production plant presented in Biegler et al. (1997).

The flowsheet and problem definition is presented below.

```
FlowSheet Ammonia

      PARAMETERS
      PP    as CalcObject(Brief="Physical Properties", File="vrpp");
      NComp as Integer;

      SET
      PP.Components = ["hydrogen", "nitrogen", "argon", "methane",
"ammonia"];
      PP.LiquidModel = "APR";
      PP.VapourModel = "APR";
      NComp = PP.NumberOfComponents;

      DEVICES
      FEED   as streamTP;
      C101   as Compressor;
      R101   as Reactor;
      F101   as flash_Steady;
      F102   as flash_Steady;
      S101   as splitter;
      M101   as Mixer;
      M102   as Mixer;
      C102   as Compressor;

      VARIABLES
      purity as fraction(Brief="Purity of the product");
      production as flow_mol(Unit = "lbmol/h", Brief="Ammonia in the
product");
      loose as flow_mol(Unit = "lbmol/h", Brief="Ammonia in the
purge");
      Q1 as heat_rate;
      Q2 as heat_rate;
```

```
      CONNECTIONS
      FEED           to   M101.Inlet1;
      M101.Outlet    to   C101.Inlet;
      C101.Outlet    to   M102.Inlet1;
      M102.Outlet    to   R101.Inlet;
      R101.Outlet    to   F101.Inlet;
      F101.OutletL   to   F102.Inlet;
      F102.OutletV   to   M101.Inlet2;
      F101.OutletV   to   S101.Inlet;
      S101.Outlet1   to   C102.Inlet;
      C102.Outlet    to   M102.Inlet2;
      Q1             to   F101.Q;
      Q2             to   F102.Q;

      SET
      R101.comp = 2; # Key component of the reaction
      R101.stoic = [-3, -1, 0, 0, 2]; # Stoichiometry of the reaction

      SPECIFY
      FEED.F = 2000 * "lbmol/h";
      FEED.T = (27 + 273.15) * "K";
      FEED.P = 10 * "atm";
      FEED.z = [0.74, 0.24, 0.01, 0.01, 0.0];
      C101.Outlet.P = 200 * "atm";
      C102.Outlet.P = 200 * "atm";
      R101.X = 0.4; # Convertion of the reactor
      F101.OutletV.P = 199 * "atm";
      F101.OutletV.T = (-34 + 273.15) * "K";
      F102.OutletV.P = 10 * "atm";
      F102.Q = 0 * "kJ/h";
      S101.frac = 0.78; # Recycle fraction

      EQUATIONS
      production = purity * F102.OutletL.F;
      purity = F102.OutletL.z(5);
      loose  = S101.Outlet2.F * S101.Outlet2.z(5);

      OPTIONS
      mode = "steady";
      outputLevel = "all";
  end

Optimization AmmoniaOPT as Ammonia

      MINIMIZE
      loose / "lbmol/h" ;

      FREE
      S101.frac;

      EQUATIONS
      loose > 0 * "lbmol/h";
      production > 90 * "lbmol/h";

      OPTIONS
      outputLevel = "all";
      relativeAccuracy = 1e-4;
end
```

In this optimization problem, first it is defined a flowsheet, named “Ammonia”, and then an optimization problem that uses the equations of this previously defined flowsheet, named “AmmoniaOPT”.

The sections “PARAMETERS”, “OPTIONS”, “VARIABLES”, “SPECIFY” and “SET” are almost the same as the previous example.

Since in this flowsheet a complete process is modeled, the section “DEVICES”, declares several equipments modeled previously in other files.

In the “CONNECTIONS” section, the connections of the equipments are made, connecting a previously defined output port with an input port.

In the “EQUATIONS” section, extra equations are defined to be used in the optimization. It is important to notice that for each new variable added it is necessary to add an equation or a specification. These equations will be used further in the optimization problem.

The proposed objective function is to minimize the Ammonia loss in the purge and it is defined in the “MINIMIZE” section. Since the degree of freedom of the “Ammonia” flowsheet is already zero, it is necessary to free, at least, one variable in order to solve the problem, this is done in the “FREE” section. The chosen free variable is the recycled split fraction. It was also added two inequality constraints to the problem in the “EQUATION” section: the loss limit that must be less than 1 lbmol/h; and the Ammonia liquid flow rate in the second flash separator that must be greater than 90 lbmol/h.

Basically, two optimizations were performed: the first one using the feasible result from a previously steady-state simulation and the second one starting from an unfeasible initial guess. Within the desired relative tolerance for the constraints an objective function (10^{-3} and 10^{-4}), only the case where a feasible initial guess is provided to the OS converged, and with tolerances tighter than 10^{-4}, none of the cases converged. The optimization performances results are presented in Table 3 while the results itself are in Table 4.

Table 3: Performance results of the optimization for the Ammonia plant problem.

Run	Absolute Tolerance NLP only	Initial guess provided by NLA	Converged	Time (s) NLP only	Iterations	Function evaluations (objective and constraints)	Objective Jacobian Evaluations	Constraints Jacobian Evaluations
1	1.0E-3	No	No	-	-	-		-
2	1.0E-3	Yes	Yes	2.0	9	16	11	11
3	1.0E-4	No	No	-	-	-	-	-
4	1.0E-4	Yes	Yes	2.8	23	30	25	25

Table 4: Results of the optimization for the Ammonia plant problem.

Run	Loose (lbmol/h)	Production (lbmol/h)	Recycled flash fraction	Purity
Initial guess	1.035	97.99	0.78	0.9979
1	-	-	-	-
2	0.277	100.57	0.943	0.9979
3	-	-	-	
4	0.277	100.57	0.943	0.9979

Again, the use of sparse algebra reduced significantly the computational cost of the problem, since the full Jacobian matrix of the equality constraints has 17956 elements (134 equations and 134 variables), and the sparse Jacobian matrix has only 581 nonzero elements, almost 30 times less elements.

3. Conclusions

Considering the problem description itself, it is also possible to notice that the use of the equipments library reduces significantly the optimization problem definition.

Although only two relatively small problems were executed, it was possible to evaluate the optimization performance. By the use of sparse algebra functionalities presented in EMSO and IPOPT, the problem size, and indirectly the computational time, could be significantly reduced. It is also necessary to simulate and optimize examples with more variables and equations.

The main advantage of the use of OS integrated with the simulation tool compared with commercial optimizers (as GAMS, or AMPL) is the simulation of separation process, since the simulation requires the calculation of thermodynamic equilibrium

data which equations are very difficult to be written. After the exhaustive test with more examples more functionalities for EMSO are planned for this year, such as: an intuitive graphical interface (where the equations/equipments can be connected like blocks), parameter estimation and the ability to solve mixed-integer non-linear optimization (MINLP) problems. For the next two years other developments are planned for EMSO, such as: data reconciliation, dynamic optimization, integration between Matlab and Scilab, CAPE-OPEN interfaces and others.

Acknowledgements

The authors would like to thanks FINEP and the ALSOC consortium industries for sponsoring of this work at Federal University of Rio Grande do Sul.

References

Banks, J. (2001). Panel session: Education for simulation practice - five perspectives. *In: Proceedings of the 2001 Winter Simulation Conference*, New Jersey, USA, 1571.

Biegler, L. T., Grossmann, I. E., Westerberg, A. W. (1997). Systematic Methods of Chemical Process Design. *Prentice Hal.*, Englewood Cliffs, NJ.

Gani, R., Ruiz, C. A., Cameron, I. T. (1986). A generalized model for distillation columns – I Model descriptions and applications. *Comp. and Chem. Eng.*,10, 181.

Grossmann, I., Biegler, L. (2004). Part II – Future Perspectives on optimization. – *Comp. and Chem. Eng.*, 28, 1193.

Hertz, D. W. (1995). Managing the design process. In: Rossiter, A. P. Waste minimization through process design. *McGraw-Hill*. New York, 265.

Hock, W., Schittkowski, K. (1981). Test Examples for Nonlinear Programming Codes. *Lect. Notes in Economics and Mathematical Systems.*, 187.

Meza, J. C., Oliva, R. A., Hough, P. D., Williams, P. J. (to appear). "OPT++: An Object-Oriented Class Library for Nonlinear Optimization", *ACM Transactions on Mathematical Software*.

Nance, R. E. (2000). Simulation education: past reflections and future directions. *In: Proceedings of the 2000 Winter Simulation Conference*. Orlando, USA, 1595.

Nance, R. E., Balci, O. (2001). Thoughts and musings on simulation education. *In: Proceedings of the 2001 Winter Simulation Conference*. Arlington, USA, 1567.

Ondrey, G. (2005). Simulation and modeling spread their wings. *Chem. Eng.*, 107, 27.

Rao, S. S. (1996). Engineering optimization : theory and practice. *John Wiley*, New York.

Soares, R.P., A.R. Secchi (2003). EMSO: A New Environment for Modeling, Simulation and Optimization. In: *Proceedings of the 13th European Symposium on Computer Aided Process Engineering* (ESCAPE 13), Lappeenranta, Finlândia, 947-952.

Tu, H., Rinnard, I. H. (2006). Foresee – A hierarchical dynamic modeling and simulation system of complex processes. *Comp. and Chem. Eng.*, 30, 1324.

Vin, L. J., Jägstam, M. (2001). Why we need to offer a modeling and simulation engineering curriculum. *In: Proceedings of the 2001 Winter Simulation Conference*, Arlington, USA ,1599.

VRTech Tecnologias Industrias Ltda (2005). *VRTherm*. Porto Alegre, Brazil.

Wächter, A. (2002). An Interior Point Algorithm for Large-Scale Nonlinear Optimization with Applications in Process Engineering, *Phd Thesis*, Carnegie Mellon University., Pittsburgh, USA.

Wächter, A., Biegler, L. T. (2006). On the Implementation of a Primal-Dual Interior Point Filter Line Search Algorithm for Large-Scale Nonlinear Programming, *Mathematical Programming*, 106, 25-57.

Yang, T. A. (2001). Integration of computer simulation and visualization research into undergraduate degree programs. *In: Proceedings of the 2001 Winter Simulation Conference.* Arlington, USA, 1592.