C2

GLOBAL OPTIMIZATION OF DYNAMIC PROCESSES: METHODS, IMPLEMENTATION AND APPLICATIONS

Carmen G. Moles, Antonio A. Alonso and Julio R. Banga*
Chemical Engineering Lab, **IIM-CSIC**
C/Eduardo Cabello 6, 36208 Vigo, Spain

Abstract

*The **optimization** of **nonlinear dynamic systems** is a very challenging task due to its frequent non convex nature. Here, several **global optimization (GO)** methods are used to effectively solve these problems with reasonable computational effort. A framework for the implementation of these GO methods with dynamic simulation models created using **EcosimPro** is described. Guidelines for the easy development of parallel versions are also given. The capabilities and performance of the resulting tool are illustrated considering the solution of two case studies (an integrated design and a parameter estimation problem).*

Key words: global optimization, dynamic optimization, parameter estimation, optimal control, integrated design, EcosimPro, Matlab

1 INTRODUCTION

The process industries (chemicals, biotechnology, food, pharmaceuticals, etc.) are under great pressure to increase efficiency, quality, safety, and minimize environmental impact. Many of their operations are performed in batch or semi-continuous mode, so they have an intrinsic dynamic nature. To that purpose, modern *process systems engineering*, as many other engineering and scientific disciplines, makes extensive use of dynamic simulation. However, dynamic simulation is rarely an end in itself. Rather, it is frequently used as the basis for systematic and rational decision making in issues like process design, operation and control.

The *mathematical optimization* framework allows a rigorous formulation of those problems. Since most processes must be modeled taking into account their dynamic nature, the optimization of systems described by non-linear differential-algebraic equations (DAEs) has become a very hot research topic during the last decade. These problems can be ultimately formulated as *non-linear programming* (NLP) problems subject to DAEs and possibly other inequality constraints, leading to the so-called *NLP-DAEs* formulation.

In the context of process engineering, the numerical solution of NLP-DAEs is usually a very challenging task due to the highly non-linear and frequently discontinuous nature of the dynamics of most processes. In fact, these characteristics make the NLP multimodal, i.e. there is not a unique global solution. Instead, a number of local solutions are possible, so the problem becomes much more complicated since the absolute best (global) solution must be found in a reliable and efficient way. In fact, standard gradient-based techniques for NLPs (e.g. SQP) are of local nature, i.e. they will converge to one of the local solutions, and in fact without giving any information about its local nature. In order to surmount these difficulties, the so-called ***Global Optimization* (*GO*)** methods must be used in order to ensure proper convergence to the truly best possible solution.

2 GLOBAL OPTIMIZATION OF DYNAMIC PROCESS

The most relevant problems regarding the optimization of dynamic processes can be roughly classified into three groups:

- *Parameter estimation problems*, also called inverse problems, or model calibrations. Here the objective is to compute the values of model parameters so the best fit between experimental data and model predictions is obtained. This type of problems is always present in the development of a dynamic model, but many users are not aware of its frequent non convexity.
- *Simultaneous design and control optimization* (i.e. integrated process design) problems. Here

* Corresponding author, julio@iim.csic.es

the objective is to simultaneously find the static variables of the process design, the operating conditions and the control system settings which optimize a combined measure of the plant economics and its controllability, subject to a set of constraints which ensure appropriate dynamic behavior and process specifications.

- *Optimal control problems*, where the aim is to compute the optimal operating policies (control variables) of a dynamic process in order to minimize (or maximize) a certain performance index, which might be a measure of e.g. economic and/or quality criteria. Optimal control problems are also relevant in many other areas, from aircraft guidance to electric power systems, business management, etc.

The problems of parameter estimation and integrated process design can be directly formulated as NLP-DAEs. Optimal control problems are originally infinite dimensional (one must find a control time-dependent function to optimize a functional), but they can be transformed to NLP-DAEs via the use of direct transformation methods (e.g., control vector parameterization).

A general statement of the corresponding global optimization problems could be stated as find **v** to minimize (or maximize)

$$\boldsymbol{C} = \boldsymbol{c}(\boldsymbol{v}, \boldsymbol{x}, \boldsymbol{p}) \tag{1}$$

subject to

$$\begin{aligned} \boldsymbol{f}(\boldsymbol{dx}/\boldsymbol{dt}, \boldsymbol{x}, \boldsymbol{v}, \boldsymbol{p}) &= 0 \\ \boldsymbol{x}(t_0) &= \boldsymbol{x}_0 \\ \boldsymbol{h}(\boldsymbol{x}, \boldsymbol{p}, \boldsymbol{v}) &= 0 \\ \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{p}, \boldsymbol{v}) &\le 0 \\ \boldsymbol{v}^L \le \boldsymbol{v} &\le \boldsymbol{v}^U \end{aligned} \tag{2}$$

where **v** is the vector of decision variables (static or dynamic), ***C*** is the objective function to minimize (or maximize), **f** is the set of differential and algebraic equality constraints describing the system dynamics (i.e. the nonlinear process model), and **h** and **g** are possible equality and inequality path and/or point constraints which express additional requirements for the process performance. Finally, $\mathbf{v}^L$ and $\mathbf{v}^U$ are the upper and lower bounds for the decision variables.

The formulation above is that of a non-linear programming problem (NLP) with differential-algebraic (DAEs) constraints. As it was mentioned, due to the highly nonlinear, constrained and possibly discontinuous and/or non-differentiable nature of the system dynamics, these problems are very often multimodal (non-convex). Therefore, if this NLP-DAEs is solved via standard NLP methods (such as Sequential Quadratic Programming, SQP, or quasi-Newton methods) it is very likely that the solution found will be of local nature. Clearly, one must use global optimization (GO) methods to ensure a proper solution. Unfortunately, solving GO problems is a non trivial exercise, and the current state of the art is far from being fully satisfactory. However, several types of GO methods can find suitable near-optimal solutions in many instances, as it will be discussed in the next section.

3 GLOBAL OPTIMIZATION METHODS

Basically, global optimization methods can be roughly classified as deterministic [8, 12, 18] and stochastic strategies [2, 3, 10, 22]. It should be noted that, although deterministic methods can guarantee global optimality for certain GO problems, no algorithm can solve general GO problems with certainty in finite time [10]. In fact, although several classes of deterministic methods (e.g. branch and bound) have sound theoretical convergence properties, the associated computational effort increases very rapidly (often exponentially) with the problem size. In contrast, many stochastic methods can locate the vicinity of global solutions with relative efficiency, but the price to pay is that global optimality can not be guaranteed.

However, in practice the process designer can be satisfied if these methods provide him/her with a very good (often, the best available) solution in modest computation times. Furthermore, stochastic methods are usually quite simple to implement and use, and they do not require transformation of the original problem, which can be treated as a black box. This characteristic is especially interesting since very often the process engineer must link the optimizer with a third-party software package where the process dynamic model has been implemented.

The GO methods that we have considered are:

- ICRS: a stochastic GO method presented by Banga and Casares [3], improving the Controlled Random Search (CRS) method of Goulcher and Casares[6]. Basically, ICRS is a sequential (one trial vector at a time), adaptive random search method which can handle inequality constraints via penalty functions.

- LJ: another simple stochastic algorithm, described by Luus and Jaakola [16]. LJ considers a population of trial vectors at each iteration. Constraints are also handled via penalty functions.

- DE: the Differential Evolution method, as presented by Storn and Price [21]. DE is a heuristic, population-based approach to GO. The original code of the DE algorithm did not check if the new generated vectors were within their bound constraints, so we have slightly modified the code for that purpose.

- GLOBAL: this is a hybrid GO method by Csendes [5], which essentially is a modification of the algorithm by Boender et al. [4]. This method uses a random search followed by a local search routine. Initially, it carries out a clustering phase were the Single Linkage method is used. Next, two different local search procedures can be selected for a second step. The first (LOCAL) is an algorithm of Quasi-Newton type that uses the DFP (Davison-Fletcher-Powell) update formula. The second, more appropriate for problems with discontinuous objective functions or derivatives, is a robust random search method (UNIRANDI) by Jarvi [14].

- MCS: the Multilevel Coordinate Search algorithm by Huyer and Neumaier [13] is intermediate between purely heuristic methods and those that allow an assessment of the quality of the minimum obtained. It has an initial global phase after which a local procedure based on a SQP algorithm is effectuated. These local enhancements lead to quick convergence once the global step has found a point in the basin of attraction of a global minimizer.

- GCLSOLVE: a deterministic GO method, implemented in Matlab as part of the optimization environment TOMLAB [11]. It is a version of the DIRECT algorithm [15] that handles nonlinear and integer constraints. GCLSOLVE runs for a predefined number of function evaluations and considers the best function value found as the global optimum.

4 IMPLEMENTATION WITH ECOSIM

A key element for the proper and efficient implementation of NLP-DAEs problems is the computational environment to develop, test and maintain the dynamic models, i.e. the DAEs. EcosimPro® is a systems simulator capable of dealing directly with DAEs. It also has a very intuitive modelling language (EL) that simplifies its usage and understanding.

The object-oriented nature of EL allows the development of models with reusable components that can be easily connected in different ways. Furthermore, EcosimPro is capable of generating C++ code that can be easily ported to any other environment. Indeed, this last characteristic was found particularly useful in this work, greatly simplifying the implementation of the dynamic models and their integration with the optimization solvers, which were implemented under the Matlab® environment.

The main reason to use Matlab was that it is a convenient environment to postprocess and visualize all the information arising from the optimization runs of the different solvers, allowing careful comparisons with little programming effort. Further, new methods (or modifications to existing ones implemented in Matlab) can be easily prototyped and evaluated. However, as a drawback, it is well known that Matlab programs usually are one order of magnitude (or more) slower than equivalent compiled Fortran or C codes. In order to minimize this effect, we have implemented the more costly part of the problems (i.e. system dynamic simulation plus objective function and constraints evaluation) in compiled C++ modules by way of EcosimPro, which are callable from the GO solvers via simple gateways. Since most stochastic methods use 90% (or more) of the computation time in system simulations (especially if their complexity level is medium to large), this procedure ensures good efficiency while retaining the main advantages of the Matlab environment.

Matlab provides an Application Program Interface (API) to interact with data and programs external to Matlab (external interfaces). The functions supported by the API included:

- Calling C or FORTRAN programs (from Matlab)
- Importing and exporting data to or from the Matlab environment.
- Establishing client/server relationships between Matlab and other software programs.

Matlab callable C and FORTRAN programs are referred as MEX-files, which are dynamically linked subroutines that the Matlab interpreter can automatically load and execute. The gateway routine needed to create the MEX file communicates the C++ code generated by EcosimPro with Matlab.

The procedure we have devised for the implementation and solution of NLP-DAEs problems involves two simple steps and is outlined in Figure 1:

i) In a first step, the user implements the system dynamics using the high level EL language in the EcosimPro environment. Then, the corresponding C++ code is created automatically. This simulation code, plus a suitable gateway code, is then used to create a dynamic link library (DLL) which can be

called directly from the Matlab (or other) environments.

ii) In a second step, the user implements the objective function and the inequality constraints to be used by the GO or local NLP solvers.

At this point, optimization computations can be performed. Since GO is an open problem, it is advisable to use different solvers for each problem, since no solver can find the global solution with absolute certainty in finite time. In this context, the Matlab environment provides excellent tools for the post-processing of the convergence information of the different solvers.

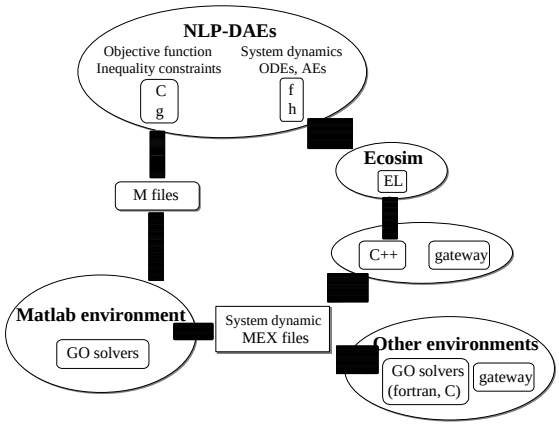


Figure 1 : Scheme of the GO implementation.

5 PARALLEL IMPLEMENTATIONS

Although stochastic GO methods do not introduce a large computational overhead, they usually require a large number of function evaluations, which usually means a large number of dynamic simulations (i.e. solution of the DAEs). Obviously, this can be very expensive for large, complex processes, even though these methods do not require computation of dynamic sensitivities. In order to surmount this drawback, the fact that several of the methods mentioned above lend themselves very well to parallelization can be exploited to keep computation times within reasonable values.

Following the current trends in **cluster computing** of using available local networks of low-cost platforms, a concept which became very popular since the Beowulf project, we have developed parallel prototypes for some of the stochastic methods which are able to run on heterogeneous clusters of PCs and workstations with a common file system. These prototypes have the following main characteristics:

- they make use of the master-worker paradigm, with the code parallelizations based on the PARALIZE Matlab tools by Abrahamsson[1], which we have found very useful.
- the monitoring and administration of the cluster is performed from the master machine with the help of the Virtual Network Computing (VNC) system by Richardson et al. [19].

VNC is a free, small (yet powerful) remote display (i.e. thin client) system which we have found particularly suitable since it is platform-independent (see www.uk.research.att.com). Using these components, we were able to develop parallel versions of the stochastic codes with very little effort. It should be noted that a more rigorous approach would be to use the PVM or MPI interfaces, but this would mean a dramatic increase of the development time, which we did not find justified for the purposes herein.

6 CASE STUDIES

The global optimization environment described previously was used to solve two case studies:

- the *integrated design and control optimization* of a wastewater treatment plant model
- the *parameter estimation* of a Global CO_2 dynamic model for planet Earth

6.1 A WASTEWATER TREATMENT PLANT

This case study represents an alternative configuration of a real wastewater treatment plant placed in Manresa (Spain), as described by Gutiérrez and Vega [9]. The plant consists of two aeration tanks, acting as bioreactors, and two settlers, as depicted in Figure 2. A flocculating microbial population (biomass) is kept inside each bioreactor, transforming the biodegradable pollutants (substrate), with the aeration turbines providing the necessary level of disolved oxygen. The effluents from the aeration tanks are separated in their associated settlers into a clean water stream and a activated sludge, which is recycled to the corresponding aeration tank. Since the activated sludge is constantly growing, more is produced that can be recycled to the tanks, so the excess is eliminated via a purge stream (qp). The objective of the control system is to keep the substrate concentration at the output (s_2) close to a given admissible value. The main disturbances come from large variations in both the flowrate and substrate concentration (q_i and s_i) of the input stream. Although there are several possibilities for the manipulated variable, here we have considered the flowrate of the sludge recycle to the first aeration tank, as considered by Gutiérrez and Vega [9].

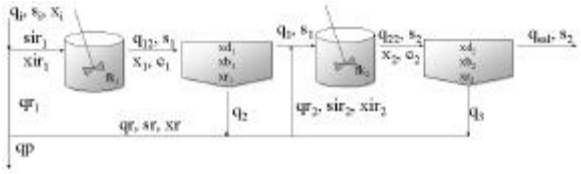


Fig 2: Wastewater treatment plant scheme

The overall dynamic model consists of 33 DAEs (14 of them are ODEs) and 44 variables. The value of three flowrates (qr_2, qr_3 and qp) are fixed at their steady-state values corresponding to a certain nominal operational conditions. Therefore, this leaves 8 design variables for the integrated design problem, namely the volume of the aeration tanks (v_1 and v_2), the areas of the settlers (ad_1 and ad_2), the aeration factors (fk_1 and fk_2), the gain and the integral time of a PI controller.

The objective is to design the plant taking into account, simultaneously, both the associated economic costs (capital and operation), essentially static measures, and the controlability of the plant, an index of its dynamic performance. This *integrated design and control problem* is formulated as an NLP-DAEs and is handled by solving the differential-algebraic equations (equality constraints), for each function evaluation, using a suitable initial value problem (IVP) solver (e.g. DASSL, as included in EcosimPro). The *objective function* to be *minimized* is a weighted sum of economic (ϕ_{econ}) and controllability cost terms (measured here as the ISE):

$$C = w_1 \cdot ISE + f_{econ} = w_1 \cdot ISE + (w_2 \cdot v_1^2) + (w_3 \cdot v_2^2) + (w_4 \cdot ad_1^2) + (w_5 \cdot ad_2^2) + (w_6 \cdot fk_1^2) + (w_7 \cdot fk_2^2) \quad (3)$$

where the ISE is the integral square error, $ISE=\int e^2(t)\cdot dt$. The ISE is evaluated considering a step disturbance to the input substrate concentration, *si*, whose behaviour is taken from the real plant. The minimization is subject to several sets of constraints:

- the 33 model DAEs (system dynamics), acting as differential-algebraic equality constraints.
- 16 inequality constraints which impose limits on the residence times and biomass loads in the aeration tanks, the hydraulic capacity in the settlers, the sludge ages in the decanters, and the recycles and purge flow rates respectively.
- an additional set of 30 double inequality constraints (representing acceptable upper and lower bounds) for the state variables.

The problem was solved with all the GO methods described previously. The best result was obtained with the DE method, which converged to C*=1538.41 after 32 minutes of computation using a PC/Pentium III (see [17] for more details). In order to provide with a more fair comparison of the different methods, a plot of the convergence curves (objective function values versus computation times) is presented in figure 3, where average curves per method are plotted. It can be seen that the ICRS method presented the most rapid convergence, but was ultimately surpassed by DE. It can also be seen that for a computation time of 200 seconds, several methods had arrived to objective function values reasonably close to the best solution. This is a common feature of stochastic GO methods: in fact slightly relaxing the tolerance for the desired solution is exponentially efficient.

In contrast, the SQP method (CONSTR code implementation, from the Matlab Optimization toolbox, [7]) converged to a bad local solution (C=3495.7), illustrating the need of using GO methods for the reliable solution of these problems as discussed previously.

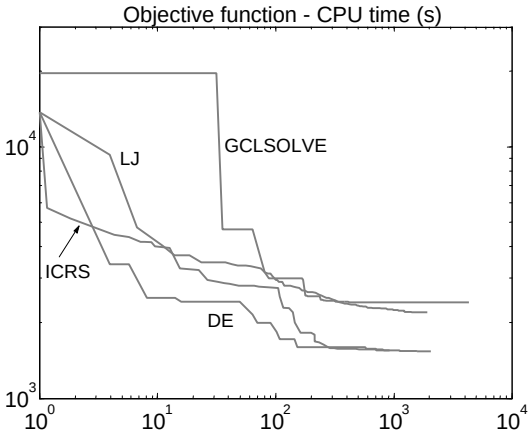
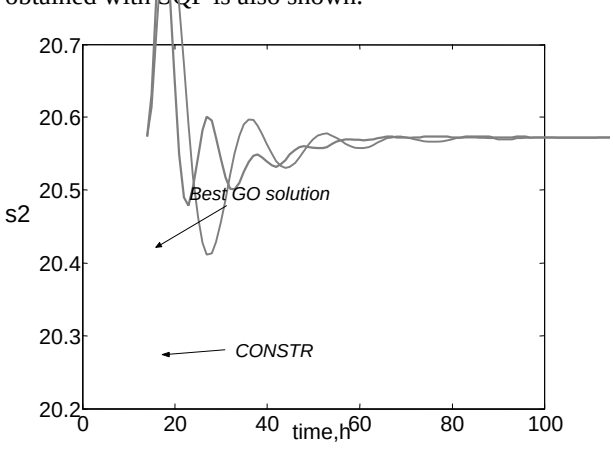


Figure 3: Convergence curves

In order to illustrate the quality of the solution obtained by the best GO method, plots for the controlled and the manipulated variables are presented in figure 4, where the local solution obtained with SQP is also shown.



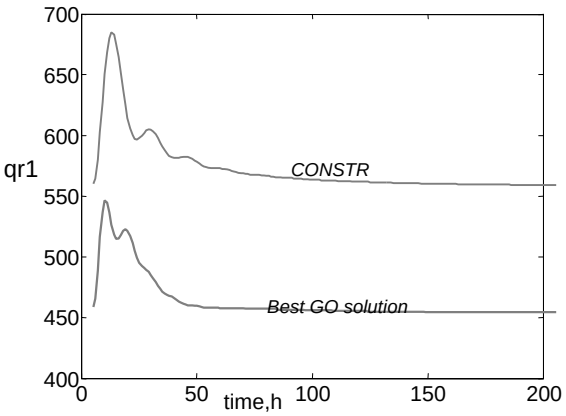


Figure 4: Controlled and manipulated variables

6.2 GLOBAL CO_2 MODEL (GCM)

The growing accumulation of CO_2 in the atmosphere due to human activity constitutes a large scale perturbation on the Earth's environment with unknown long-term consequences. The global CO_2 model (GCM) is a low-order ODE model which is intended as an introduction to the global CO_2 problem. The Earth's environment is modeled as seven completely mixed reservoirs. This simplified model (details are given in [20]) describes the generation and accumulation of CO_2 in our planet as a result of the interaction of seven reservoirs (see figure 5), namely the upper atmosphere, the lower atmosphere, the long-lived biota, the short-lived biota, the mixed ocean layer, the deep sea and finally the marine biosphere. The overall dynamic model consists of 15 DAEs (7 of them are ODEs)

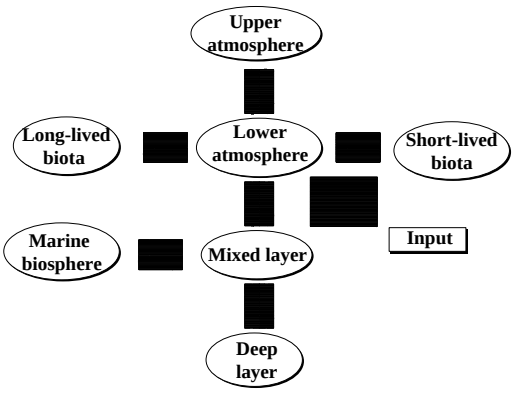


Figure 5: Interactions of the reservoirs in the GCM model

In this work, we have used this model to formulate a **parameter estimation** (model calibration) problem. The aim was, given a set of available data (*m* measures of *n* state variables during a certain time horizon), to estimate ten parameters of the model which give the best fit between model predictions and the real data. These parameters represent the lower atmosphere to ocean mixed layer exchange time, the upper to lower atmosphere exchange time, the ocean mixed layer to deep sea exchange time, the biota growth factor and the ocean mixed layer evasion factor for excess carbon dioxide. Furthermore, the five last parameters correspond to ratios with respect to the total carbon in the lower atmosphere prior to the industrial era.

The optimization problem (parameter estimation of a DAEs system) is stated as the minimization of a weighted distance measure between experimental and predicted values of a subset of state variables:

$$f = \sum_{i=1}^{n}\sum_{j=1}^{m} w_{ij}\left((y_{teor}(i) - y_{\exp}(i))_j\right)^2 \qquad (4)$$

where y_{exp} represents the known experimental data, that is, the variation in the carbon concentration experimented at each reservoirs for the last 300 years (from 1700 to 2000), while y_{teor} corresponds to the predicted theoretical evolution using the model with a given set of the ten parameters.

The usual approach to solve parameter estimation problems is to use the Levenberg-Marquardt (L-M) method. The main drawback of L-M method (e.g. as implemented in the Matlab Optimization toolbox, [7]) arises from the fact that it is a local gradient method, thus it may converge to local solutions depending on the initial point. In fact, this was our experience when using it to solve the above stated problem (i.e. lack of robustness).

This important shortcoming can be properly surmounted by using suitable GO methods. Here, the best result was achieved by the ICRS method, closely followed by the DE method. Both methods arrived to objective function values several orders of magnitude better than the standard L-M method (see table 1). The results presented in this table also indicate a common characteristic of many parameter estimation problems: there is a low sensitivity of the objective function with respect to some decision variables, i.e. rather different decision vectors might have almost the same dynamic response with respect to the experimental data considered. In practice, this property creates severe ill-conditioning and/or multimodality, so gradient based methods usually fail, as it was found here.

Regarding the best solution obtained (using ICRS), plots for the CO_2 concentration in the upper and lower atmosphere respectively are given in figure 6, showing a very good agreement between the experimental and predicted values. But it should be noted that even with this solver, there are significant

differences between the obtained and the nominal values of the model parameters, which is again a consequence on the ill-conditioning of the problem and possibly of the identifiability of these parameters. In any case, GO methods offer increased robustness and reliability with respect to standard gradient approaches.

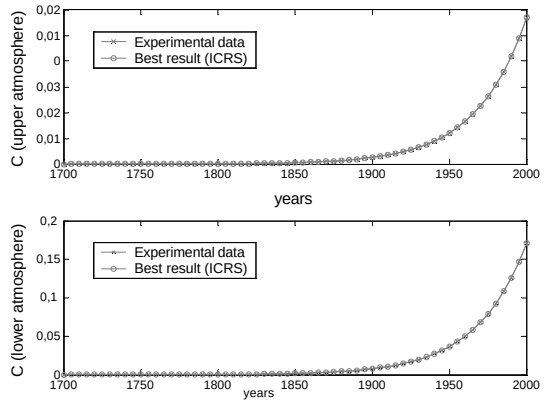


Figure 6: Evolution of the CO_2 concentration (as relative carbon increase) in the atmosphere: experimental vs model predictions

Table 1: Nominal values of the parameters and results obtained with selected global and local approaches

Parameter name	Nominal values	ICRS	DE	LEASTSQ (Lev-Marq)
BETA	0.6	0.599	0.599	0.986
TUL	2.0	119.49	1.876	1.575
TAM	5.8	41.618	11.573	1.018
TDM	1500.0	1519.3	1546.1	899.9
EPS	12.5	14.973	11.668	5.035
P1	0.15	0.437	0.150	0.048
P2	2.534	29.382	3.036	0.993
P3	0.122	0.186	0.129	0.286
P4	2.0	1.977	1.978	14.986
P5	0.1	0.140	5.890	4.999
F. obj. (4)	-	***5.0e-8***	*6.9e-7*	*0.1439*

7 CONCLUSIONS

The optimization of nonlinear dynamic systems is a very challenging task due to its frequent multimodal nature. We have shown how several global optimization (GO) methods can be effectively used to solve these problems with reasonable computational effort. We have also described a framework for the implementation of GO methods using a combination of the Matlab environment with dynamic simulation models created using EcosimPro. In addition, guidelines for the easy development of parallel versions were devised and successfully tested. The application of the resulting tool to the solution of two case studies (an integrated design and a parameter estimation problem) were used to illustrate its capabilities and performance.

8 ON-GOING AND FUTURE WORK

This framework is being used to implement and solve a large number of optimization problems, including the integrated design and optimal control of large-scale dynamic systems. Examples of these systems are those arising from the discretization of distributed processes. So far, optimization problems with more than 5000 differential states have been solved. Updated information of our recent and current activities in this field can be found at:
`http://www.iim.csic.es/~julio/`

Acknowledgements
We would like to thank Gloria Gutiérrez (UVA) for providing us with the wastewater plant dynamic model. Thanks also to Pedro Cobas and Luis Ordoñez (EA Int.) for their helpful hints on issues regarding EcosimPro model implementations for efficient dynamic simulation.

References

[1] Abrahamsson, T., (1998) PARALIZE. Dept. of Mechanical Engineering, Chalmers University of Technology, Sweden.

[2] Ali, M., Storey, C., Törn, A.; (1997) Application of stochastic global optimization algorithms to practical problems. J. Optim. Theory Appl. 95, 545

[3] Banga, J.R., Casares, J.J.; (1987) Integrated controlled random search: application to a wastewater treatment plant model. IChemE Symp. Ser. 100, 183.

[4] Boender, C.G.E., Rinnooy Kan, A.H.G., Timmer, G.T., Stougie, L. A.; (1982) Stochastic method for global optimization. Math. Programming. 22, 125.

[5] Csendes, T.; (1988) Nonlinear parameter estimation by global optimization - efficiency and reliability. Acta Cybernetica. 8, 361.

[6] Goulcher, R., Casares, J.J. (1978) The solution of steady-state chemical engineering optimization problems using a random search technique. Comput. Chem. Eng., 2, 33.

[7] Grace, A. (1994) Optimization Toolbox User's Guide; The Math Works Inc.

[8] Grossmann, I.E.; (1996) Global Optimization in Engineering Design. Kluwer Academic Publishers: Dordrecht.

[9] Gutiérrez, G., Vega, P. (2000) Integrated design of activated sludge process taking in to account the closed loop controllability. Proceedings of ESCAPE-10 Conference, Firenze, Suppl. Vol, p. 63-69.

[10] Guus, C., Boender, E., Romeijn, H. E.; (1995) Stochastic methods. In Horst, R.; Pardalos, P.M. eds. Handbook of Global Optimization. Kluwer Academic Publishers.

[11] Holmström, K.; (1999) The TOMLAB optimization environment in Matlab. Adv. Model. Optim., 1 (1), 47.

[12] Horst, R., Tuy, H.(1996) Global Optimization - Deterministic Approaches. Springer-Verlag, 3rd Edn.: Berlin.

[13] Huyer, W., Neumaier, A.; (1999) Global Optimization by Multilevel Coordinate Search. J. Global Optimization 14 , 331-355.

[14] Jarvi, T. (1973) A random search optimiser with an application to a maxmin problem. Publications of the Inst. of Appl. Math., Univ. of Turk., 3.

[15] Jones, D. R. (2001) DIRECT. In Encyclopedia of Optimization, to be published.

[16] Luus, R., Jaakola, T.H.I.; (1973) Optimization by direct search and systematic reduction of the size of search region. AIChE J., 19, 760.

[17] Moles, C.G.; Gutiérrez, G.; Alonso, A.A.; Banga, J.R. (2001) Integrated process design and control via global optimization: a wastewater treatment plant case study. Submitted.

[18] Pintèr, J.D. (1996) Global Optimization in Action. Kluwer Academic Publishers: Dordrecht.

[19] Richardson, T.Q., Stafford-Fraser, K. R., Wood, A. Hopper (1998) Virtual Network Computing. IEEE Internet Comput., 2, 33-38.

[20] Schiesser, W.E. (2001) Introductory Global CO2 Model. Available at http://www.lehigh.edu/~wes1/ co2/co2.html

[21] Storn, R., Price, K.; (1997) Differential evolution - a simple and efficient adaptive scheme for global optimization over continuous spaces. J. Global Optim., 11, 341.

[22] Törn, A., Ali, M., Viitanen, S. (1999) Stochastic global optimization: problem classes and solution techniques. J. Global Optim., 14, 437.