AIDIC *Conference Series*, Vol. 9, 2009
Editor Sauro Pierucci
DOI: 10.3303/ACOS0909024

ISBN 978-88-95608-10-5 ISSN 2036-5969

A CONSCIOUS CLASS TO HANDLE CONSTRAINED OPTIMIZATIONS AND DIFFERENTIAL SYSTEMS IN OPTIMAL CONTROL ISSUES

Flavio Manenti[1*], Guido Buzzi-Ferraris[1], Ivan Dones[2], Heinz A. Preisig[2]

[1] Dipartimento di Chimica, Materiali e Ingegneria Chimica "Giulio Natta", Politecnico di Milano
Piazza Leonardo da Vinci 32, 20133 Milano, ITALY

[2] Chemical Engineering Department, Norwegian University of Science and Technology (NTNU)
Sem Sælands vei 4, 7491 Trondheim, NORWAY

A generalized C++ class to solve nonlinear model predictive control (MPC) and dynamic optimization issues is proposed. Since optimal control problems involve (i) differential equations systems to foresee plants and/or process unit dynamics and (ii) constrained optimization issues to meet process specs and requirements, *BzzMath* library is adopted as kernel to numerically and consciously face these tasks. The proposed class allows both FORTRAN and C++ users to easily solve MPC and dynamic optimization by only defining their differential system and the desired objective function without worrying about numerical problems that may occur while integrating differential systems and searching for the minimum of a constrained/complex objective function.

1. INTRODUCTION

The twofold aim in studying a generalized class to solve optimal control problems is the need of finding an efficient solution for the supply chain management problem as well as to propose and validate a freely downloadable tool to support users in settling nonlinear model predictive control (MPC) and business-wide dynamic optimization.
Actually, it is field-proven that MPC methodology is one of the most promising approaches to ensure flexible and profitable production with an economic optimization of plant operations by ensuring the reduction of downtimes, product waste, and raw materials cost impact. It is carried out meeting process constraints and guaranteeing a reliable control system. Notwithstanding, the industrial state of art in process control is still represented by the linear MPC, even though nonlinear applications have significantly increased in number in the last five years as testified in the literature (Bauer and Craig, 2008; Qin and Badgwell, 2003). The delay in nonlinear applications is especially due to the traditional inertia of process industries in acquiring and applying new technologies and even to the lack of free tools to approach multifaceted optimal control problems. BzzMath library (Buzzi-Ferraris, 2009) offers a comprehensive and reliable numerical kernel to develop a generalized class for nonlinear MPC applications. BzzMath library is briefly presented in Section 2. Section 3 discusses MPC structure. Section 4 introduces the case study adopted to validate the class. Finally, functionalities of the proposed tool are in Section 5. Also, an appendix is attached at the end to show the easy of building MPC objects for C/C++ users, whereas FORTRAN users can refer to incoming Wiley's book by the same authors (Buzzi-Ferraris and Manenti, 2010a, 2010b).

[*] Corresponding author:
Phone: +39 02 2399 3273; Fax: +39 02 7063 8173; Email: flavio.manenti@polimi.it

Please cite this article as: Manenti F., Buzzi-Ferraris G., Dones I. and Preisig H.A., (2009), A conscious class to handle constrained optimizations and differential systems in optimal control issues, AIDIC Conference Series, 09, 205-212
DOI: 10.3303/ACOS0909024

2. BZZMATH LIBRARY AS KERNEL

This research activity is based on BzzMath library, which is a numerical library freely downloadable at Professor Buzzi-Ferraris's homepage. BzzMath is entirely written in C++ by exploiting object-oriented programming (Buzzi-Ferraris, 1994) and, in its last release, openMP directives for parallel computing on shared memory machines. It covers several scientific fields such as linear algebra, linear/nonlinear regressions, optimization, differential systems and so on, and some of them are reported hereinafter.

2.1 Linear Algebra

Linear algebra is the essential basis to solve many numerical problems:

> *"The importance of numerical linear algebra in modern scientific computing cannot be overstated"*
> *(Gill et al., 1991)*

BzzMath is predisposed not only to solve linear systems, but even to automatically adopt the most performing algorithm in accordance with the type of system to be solved. Being developed in object-oriented way, objects belonging BzzMath library can easily identify and, if possible, exploit matrix sparsity and, when sparse, the possible matrix structure of linear systems, by making their solution very performing. In addition, a relevant error that still affects other numerical libraries has recently been highlighted, demonstrated, and fixed (Buzzi-Ferraris and Manenti, 2010a).

2.2 Regression Models

Parameter estimations, outlier detections, model discriminations, and design of experiments are well-known hard problems. BzzMath includes specific classes based on very robust algorithms to solve linear and nonlinear regression problems (Buzzi-Ferraris and Manenti, 2010b) and to detect masking effects, heteroscedasticity, parameter correlations, and gross errors (Manenti and Buzzi-Ferraris, 2009). A methodology to discriminate among rival models and, at the same time, to define the optimal design of experiment is even implemented in BzzMath; correct meanings of statistical tests and confidence region have recently been redefined (Buzzi-Ferraris and Manenti, 2009a).

2.3 Nonlinear Systems and Optimization

Starting from OPTNOV's variant (Buzzi-Ferraris, 1967; Buzzi-Ferraris and Manenti, 2010, submitted) up to the most recent improvements, numerically robust and efficient algorithms are implemented for solving nonlinear systems and optimization problems. On this subject, some examples solved by means of BzzMath library can be quoted: a very large-scale nonlinear system consisting of sparse blocks matrix with a number of equations in the order of some tens of millions to characterize a kinetic post-processor (Cuoci *et al.*, 2007); a constrained multiscale and multiobjective optimization involving a diagonal block matrix representing a polyethylene terephthalate polymer plant (Manenti and Rovaglio, 2008); and a self-regulating MPC for a C3/C4 splitter to manage the model reduction and the level of detail of process dynamics characterization (Dones *et al.*, 2009).

2.4 Differential and Differential-Algebraic Systems

At last, BzzMath library includes reliable and very performing algorithms for solving ordinary differential equations (ODE) systems (Buzzi-Ferraris and Manca, 1998) and differential and algebraic equations (DAE) systems (Manenti *et al.*, 2009). Again, its object-oriented structure gives the possibility to significantly reduce the computational time in integrating differential systems, apart from their stiffness. Many applications of BzzMath solvers are proposed in the scientific literature. In addition, ad hoc solvers to tackle partially structured DAE systems, typical of process control and process systems engineering, have recently been

introduced into the numerical library (Manenti *et al.*, 2009). Such an approach is suitable when the integral part of proportional-integral or proportional-integral-derivative loops spoils the structure of the Jacobian and the solution is generally four times faster.

3. CLASS ARCHITECTURE

The MPC algorithm was developed by Cutler at Shell Oil Company in 1979. At that time it was based on linear models, and it was called dynamic matrix control (DMC) (Cutler and Ramaker, 1980). Its basic idea is to use a time-domain step-response model (called the convolution model) of the process to predict the future responses and to obtain the optimal movements of the manipulated variables to optimize the process yield and efficiency. Using this approach, one obtains the future output responses matching the optimal trajectories in the HP (prediction horizon) by finding the best values of the manipulated variables in the HC (control horizon).
This is exactly the concept of a least squares problem of fitting HP data points with HC coefficients. The aim of this predictive control is to drive future outputs close to the reference trajectory while optimizing an objective function. The computation sequence is to estimate at first the output predictions by using the model of the plant. Then, the errors between predicted and reference trajectories are calculated. The next step is to compute the sequence of the future control actions by minimizing an appropriate quadratic objective function. However, only the first control action is implemented. This procedure is iterated by means of moving the time horizon (Henson and Seborg, 1997; Morari and Lee, 1999).
In the last decades, MPC based on nonlinear model (nonlinear MPC) has started to be studied. This has mostly been motivated by the following reasons:

- nonlinear MPC is able to handle nonlinearities both in process dynamics and in economical objective functions;
- it can be based on first-principles mathematical models and on nonlinear semi-empirical models;
- it allows simultaneously solving the predictive control (quadratic optimal control problem) and the dynamic optimization (economical optimal control problem).

The basic architecture of MPC is reported in Figure 1. Assuming an on-line implementation of this technique, the plant provides data to the model predictive controller at each sampling time. Specifically, the plant data are sent to the optimization algorithm, which includes an objective function, a dynamic model and, usually (according to the mathematical model type) a numerical integrator to solve differential equation systems.
If the real-time dynamic optimization is to be solved as well, economical data and market scenarios must be provided to the MPC structure and one or more economical objective functions must be defined.

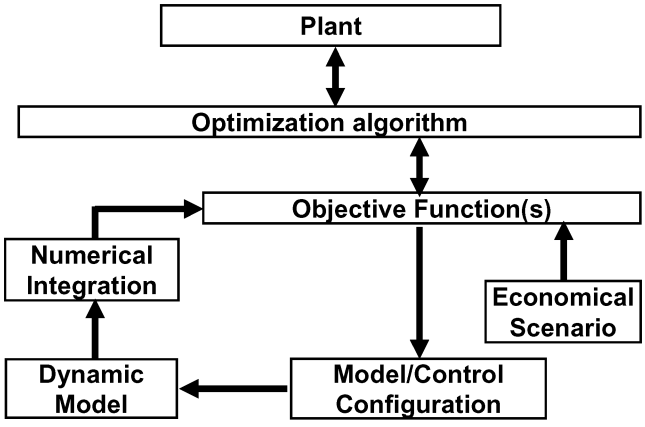

Figure 1: Architecture of model predictive control.

3.1 Objective Function

Many optimal control problems such as data reconciliation (Arora and Biegler, 2001; Bagajewicz, 2003; Manenti, 2009; Signor *et al.*, 2010), regressions (Buzzi-Ferraris and Manenti, 2009a, 2009b, 2010b), rival models (Atkinson and Fedorov, 1975; Buzzi-Ferraris and Forzatti, 1983; Hunter and Reiner, 1965), and so on can be formulated as a minimization of a least squares objective function subject to equality and/or inequality constraints. MPC enters this family and its generalized formulation is often the following one:

$$\min_{\hat{u}(k)\ldots\hat{u}(k+h_c-1)} \left\{ \sum_{j=k+1}^{k+h_p} \omega_y \left[\hat{e}_y(j) \right]^2 + \sum_{l=k}^{k+h_p-1} \omega_T \left[\hat{u}(l) - \hat{u}_{SST}(l) \right]^2 + \sum_{i=k}^{k+h_p-1} \omega_u \left[\Delta\hat{u}(i) \right]^2 \right\}$$

subject to: (1)

$$\begin{cases} u_i^{\min} \le u\left(k+i\middle|k\right) \le u_i^{\max} \\ \Delta u_i^{\min} \le \Delta u\left(k+i\middle|k\right) \le \Delta u_i^{\max} \\ -\varepsilon + y_i^{\min} \le y\left(k+i+1\middle|k\right) \le y_i^{\max} + \varepsilon \\ \Delta u\left(k+j\middle|k=0\right), \quad j=m,\ldots,p \\ \varepsilon \ge 0 \end{cases}$$

where:

- $\hat{e}_y(j) = \hat{y}(j) - y_{set}(j)$: vector of the deviations between controlled variables and their set-points;
- $\Delta\hat{u}(i) = \hat{u}(i) - \hat{u}(i-1)$: vector of incremental variations of the manipulated variables;
- ω_y , ω_T , ω_u : multiplicative coefficients, the weighting factors;
- $\hat{y}_{set}(j)$: vector of set-points of the controlled variables at the $j-th$ time-interval;
- $\hat{y}(j)$: vector of predicted controlled variables at the $j-th$ time-interval;
- $\hat{u}_{SST}(l)$: vector of the steady-state targets of the manipulated variables at the $l-th$ time-interval;
- $\hat{u}(l)$: vector of manipulated variables at the $l-th$ time-interval;
- HP : prediction horizon;
- HC : control horizon.

Despite the fact that most real-world processes are approximately linear only within a limited operating range, linear models are used in a much larger operating range.

In many cases the feedback in the controller makes the controlled system sufficiently robust to cope with these additional linearization errors. Nevertheless, if the specifications for the controlled system cannot be met anymore, one may have to use the nonlinear models, thus implement nonlinear MPC also abbreviated as NMPC. NMPC uses a nonlinear model directly in the control application; therefore the nonlinear model becomes a core component of the control application (Binder *et al.*, 2001). The nonlinear model may be in the form of an empirical data fit (*e.g.*, artificial neural networks, fuzzy models… (Lima *et al.*, 2009; Lima *et al.*, 2008)) or a high-fidelity model based on conservation laws of mass and energy (first principles model).

As in linear MPC, NMPC requires the iterative solution of an optimal control problem on a finite prediction horizon. In contrast to the linear MPC where a linear optimization problem is solved in each step, NMPC solves in every step a nonlinear problem that is computationally expensive both in the computation of the optimization criterion as well as the solution of the model (especially for large models). Due to limitations in computation time, it is therefore interesting to investigate the use of simplified nonlinear models (instead of full nonlinear models) thereby at least reducing the costs associated with the computation of the model (Dones *et al.*, 2009).

3.2 The Algorithm

The aforementioned formulation can be converted into an algorithm based on differential solvers, optimizers, and outlier detection methods belonging to BzzMath library. First of all, raw data acquired by the field should be treated in order to detect any gross error or bad quality measure.
An opportune class to reconcile raw data set based on **QR** factorization and linear systems solution already discussed and validated elsewhere (Buzzi-Ferraris and Manenti, 2010a; Signor *et al.*, 2010) is adopted. Reconciled data is then used to initialize MPC structure: the optimizer is called the first time to evaluate the best manipulated variables $\mathbf{u}$ by minimizing the objective function (1).
To do so, all equality and inequality constraints (including the differential system) have to be evaluated and an opportune differential solver should be invoked. The differential system is then integrated on a specific prediction horizon h_p in order to predict future system behavior according to different values of $\mathbf{u}$. After an iterative procedure, the optimal vector $\mathbf{u}$ is implemented in the plant and new data are acquired to restart the cycle.

4. CASE STUDY

The case study taken into consideration is a PolyEthylene Terephthalate (PET) plant. The choice of the PET process is due to its marked nonlinear behavior and to the fact that polymer plants are more and more forced to operate under frequent grade transitions to match the competitiveness of worldwide market and to satisfy the final customer demand.

4.1 The PET Plant

Figure 2 shows a generic PET production train consisting of six sections:

- a primary esterification, called PE, characterized by a CSTR (Continuous Stirred Tank Reactor), where EG (Ethylene Glycol) and TPA (TerePhthalic Acid) are fed to the reactor.
  Typically, it operates at a pressure of 1-8 bar and a temperature of 255-280 °C, lower bounded by the solubility of TPA in the oligomer.
- Downstream there is a SE (Secondary Esterification) unit, which consists of evenly-sized three chambers reactor modeled by 3 CSTRs in series.
  SE is usually run close to atmospheric conditions with temperatures a bit higher than PE.
- The following section is the LP (Low Polymerizer), with a simple CSTR, operating at a medium vacuum pressure (less than 500 Torr).
  It is possible to obtain the lowest grade end product with an intrinsic viscosity about 0.2 dl/g, which represents the polymerization index (polymer quality or polymer degree).
- Subsequently there are the IP (Intermediate Polymerizer) and the HP (High Polymerizer), both characterized by rolling discs reactors which operate at lower vacuum pressure (less than 1 Torr).
  In this case the intrinsic viscosity can be controlled through an inferential control that measures the absorbance of the electric motor.
- Downstream a SSP (Solid State Polymerizer) exploits poly-condensation phenomena to produce a very high intrinsic viscosity polymer required by tire-cord resins.

The production can vary from textile fibers, with an IV (Intrinsic Viscosity) around 0.55-0.65 dl/g, to bottles for mineral water, with 0.72-0.85 dl/g IV value, to special fibers for tire-cord resins market, where the IV can exceed 1 dl/g.

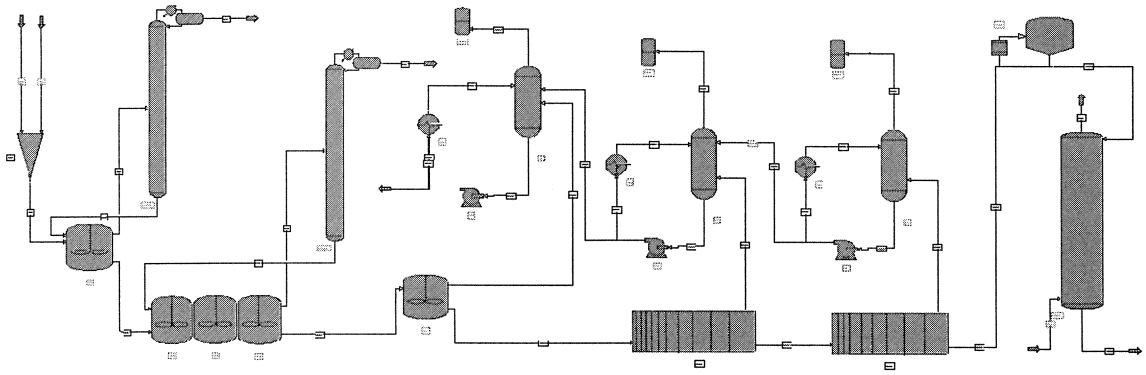

Figure 2: flow-sheet of a PET plant.

5. CLASS VALIDATION

Generalized class here proposed was validated on PET DAE model discussed by Manenti and Rovaglio (Manenti and Rovaglio, 2008). It consists of a sparse, structured Jacobian matrix, specifically of a diagonal blocks structure. Facing the well-known superior performances of the nonlinear MPC against conventional control, Figure 3 shows trends for intrinsic viscosity (IV) and pressure (P) in both the intermediate (IP) and high polymerizer (HP) during a grade change production in order to corroborate previous trends obtained by more complex procedural structure and hence to validate the generalized approach.

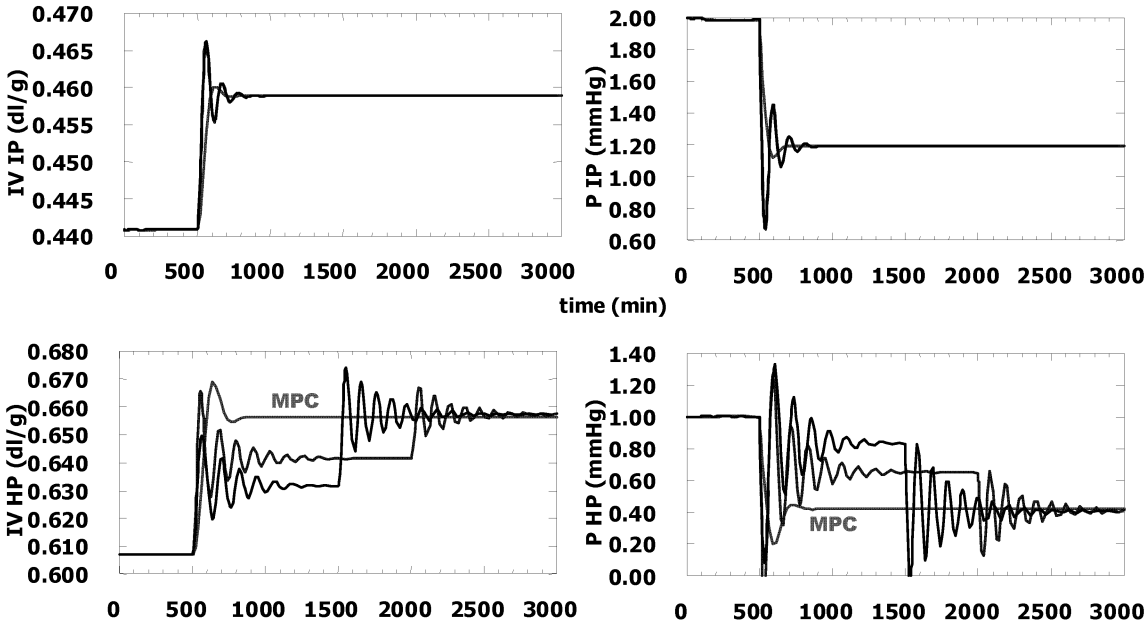

Figure 3: Nonlinear MPC applied to PET plant and compared to conventional control.

For their similar nature, MPC and dynamic optimization can be both solved by means of the generalized C++ by opportunely defining the objective function.

6. CONCLUSIONS

A generalized class for solving nonlinear MPC and dynamic optimization problems based on BzzMath library was proposed and validated on different case studies and encouraging results were obtained. Such a class allows the implementation of nonlinear MPC once an adequate objective function and a differential system are defined.
It can be easily used in FORTRAN and C++ environment without the need to worry about differential solvers and optimization algorithms, since the object-oriented nature of the class and the philosophy adopted in BzzMath library synergistically allow an automatic selection of appropriate algorithms to solve these issues.
It is worth remarking that even commercial dynamic simulators such as DYNSIM, UNISIM, ASPENHYSYS and so on could be used to develop the dynamic model and to interface it to the C++ class. In such a case, the differential solver is the one included in the dynamic simulation package and not the ones implemented in the C++ generalized class by slightly reducing the same class flexibility and operability.

7. REFERENCES

N. Arora, L.T. Biegler. (2001). Redescending estimators for data reconciliation and parameter estimation. Computers & Chemical Engineering, 25(11-12), 1585-1599.
A.C. Atkinson, V.V. Fedorov. (1975). The design of experiments for discriminating between two rival models. Biometrika, 62, 57-70.
M.J. Bagajewicz. (2003). Data Reconciliation and Instrumentation Upgrade. Overview and Challenges. FOCAPO 2003. 4th International Conference of Computer-Aided Process Operations, Coral Springs, Florida, 103-116.
M. Bauer, I.K. Craig. (2008). Economic Assessment of Advanced Process Control - A Survey and Framework. Journal of Process Control, 18, 2-18.
T. Binder, L. Blank, H.G. Bock, R. Bulirsch, W. Dahmen, M. Diehl, T. Kronseder, W. Marquardt, J.P. Schlöder, O.v. Stryk. (2001). Introduction to Model Based Optimization of Chemical Processes on Moving Horizons. In K. Groetschel, Rambau (Ed.), Online Optimization of Large Scale Systems: State of the Art: Springer.
G. Buzzi-Ferraris. (1967). Ottimizzazione di funzioni a più variabili. Nota I. Variabili non vincolate. Ing. Chim. It., 3, 101.
G. Buzzi-Ferraris. (1994). Scientific C++. Building Numerical Libraries, the Object-Oriented Way. 2nd Ed., 479pp, Addison-Wesley, Cambridge University Press, ISBN 0-201-63192-X.
G. Buzzi-Ferraris. (2009). BzzMath: Numerical library in C++. Politecnico di Milano, http://chem.polimi.it/homes/gbuzzi.
G. Buzzi-Ferraris, P. Forzatti. (1983). A new sequential experimental design procedure for discriminating among rival models. Chem. Eng. Science, 38, 225-232.
G. Buzzi-Ferraris, D. Manca. (1998). BzzOde: a new C++ class for the solution of stiff and non-stiff ordinary differential equation systems. Computers & Chemical Engineering, 22(11), 1595-1621.
G. Buzzi-Ferraris, F. Manenti. (2009a). Kinetic models analysis. Chemical Engineering Science, 64(5), 1061-1074.
G. Buzzi-Ferraris, F. Manenti. (2009b). Outlier Detection. Computers & Chemical Engineering, submitted.
G. Buzzi-Ferraris, F. Manenti. (2010a). Fundamentals and Linear Algebra for the Chemical Engineer Solving Numerical Problems. ISBN: 978-3-527-32552-8, WILEY-VCH, Weinheim, Germany.
G. Buzzi-Ferraris, F. Manenti. (2010b). Interpolation and Regression Models for the Chemical Engineer Solving Numerical Problems. ISBN: 978-3-527-32652-5, WILEY-VCH, Weinheim, Germany.
G. Buzzi-Ferraris, F. Manenti. (2010, submitted). A Combination of Parallel Computing and Object-Oriented Programming to Improve Optimizer Robustness and Efficiency. Computer Aided Chemical Engineering.
A. Cuoci, A. Frassoldati, G. Buzzi-Ferraris, T. Faravelli, E. Ranzi. (2007). The ignition, combustion and flame structure of carbon monoxide/hydrogen mixtures. Note 2: Fluid dynamics and kinetic aspects of syngas combustion. International Journal of Hydrogen Energy, 32(15), 3486-3500.

C.R. Cutler, B.L. Ramaker. (1980). Dynamic matrix control - A computer control algorithm. Proceedings of the Joint Automatic Control Conference.
I. Dones, F. Manenti, H.A. Preisig, G. Buzzi-Ferraris. (2009). Nonlinear Model Predictive Control: a Self-Adaptive Approach. Industrial & Engineering Chemistry Research, submitted.
P.E. Gill, W. Murray, M.H. Wright. (1991). Numerical Linear Algebra and Optimization. Vol. 1. Addison-Wesley, Redwood City (CA), USA.
M.A. Henson, D.E. Seborg. (1997). Nonlinear Process Control. In. Upper Saddle River, New Jersey: Prentice Hall.
W.G. Hunter, A.M. Reiner. (1965). Design for discriminating between two rival models. Technometrics, 7, 307-323.
N.M.N. Lima, F. Manenti, R. Maciel Filho, M. Embiruçu, M.R. Wolf Maciel. (2009). Fuzzy Model-Based Predictive Hybrid Control of Polymerization Processes. Industrial & Engineering Chemistry Research, 48(18), 8542–8550.
N.M.N. Lima, L. Zuniga, R. Maciel Filho, F. Manenti, D. Manca, M. Embiruçu. (2008). Dynamic Optimization of MMA and VAc Copolimerization Process. Proceedings of AIChE Meeting, Philadelphia, Pennsylvania, USA.
F. Manenti. (2009). Reacting to predicting technologies: A novel performance monitoring technique based on detailed dynamic models. Chemical Product and Process Modeling, 4(2).
F. Manenti, G. Buzzi-Ferraris. (2009). Criteria for Outliers Detection in Nonlinear Regression Problems. In J. Jezowski & J. Thullie (Eds.), Computer Aided Chemical Engineering (Vol. 26, pp. 913-917).
F. Manenti, I. Dones, G. Buzzi-Ferraris, H.A. Preisig. (2009). Efficient Numerical Solver of Partially Structured DAE Systems. Industrial & Engineering Chemistry Research, 48, 9979-9984.
F. Manenti, M. Rovaglio. (2008). Integrated multilevel optimization in large-scale polyethylene terephthalate plants. Industrial and Engineering Chemistry Research, 47(1), 92-104.
M. Morari, J.H. Lee. (1999). Model predictive control: past, present and future. Computers & Chemical Engineering, 23(4-5), 667-682.
S.J. Qin, T.A. Badgwell. (2003). A survey of industrial model predictive control technology. Control Engineering Practice, 11(7), 733-764.
S. Signor, F. Manenti, M.G. Grottoli, P. Fabbri, S. Pierucci. (2010). Robust and Reliable Data Reconciliation on Sulfur Recovery Units. Note I: Adaptive Simulation and Model Validation on Industrial Plant. Industrial & Engineering Chemistry Research, submitted.

8. APPENDIX – CLASS IMPLEMENTATION FOR C/C++ USERS

Given an objective function and a differential (or differential-algebraic) system, the MPC can be invoked through the following basic constructor:

```
BzzModelPredictiveControl NMPC(hp,hc,y0,u0,ad,DinSys,FObj,uL,uU);
```

where `hp` is the prediction horizon; `hc` the control horizon; `y0` the reconciled measures acquired by the field for each MPC call; `u0` the initial values of manipulated variables; `ad` is an integer vector for discriminating between algebraic (`ad(i) = 0`) and differential (`ad(i) = 1`) equations of the system described in the function `DinSys`; `FObj` is the weight least squares objective function; optionally, `uL` and `uU` are minimum and maximum constraints, respectively, of manipulated variables.
FORTRAN users can refer to (Buzzi-Ferraris and Manenti, 2010a, 2010b) for a detailed description and examples of C++ class implementation in FORTRAN environment.