

Computationally efficient process control with neural network-based predictive models

Luis Alberto Paz Suárez, Petia Georgieva, *Member IEEE*, Sebastião Feyo de Azevedo

Abstract— The present work reports our study on the benefits of integrating the Artificial Neural Network (ANN) technique as a time series predictor, with the concept of Model-based Predictive Control (MPC) in order to build an efficient process control. The combination of ANN and MPC usually leads to computationally very demanding procedure, that finally makes this approach less popular or even impossible to apply for real time industrial applications. The main contribution of this paper is the introduction of an error tolerance in the MPC optimization algorithm that reduces considerably the computational costs. Besides, the new ANN-MPC framework proved to bring substantial improvements compared with traditional Proportional-Integral (PI) control with respect to macro process performance measures as less energy consumption and higher productivity.

I. INTRODUCTION

Over the last 20 years, the ANNs became a well-established methodology not only as a reliable classifier with countless applications but also as dynamical regressor mainly for time-series prediction and identification. The remarkable success of the ANN approach is in great part due to the following features [1]: i) ANN are universal approximators. It has been proven that any continuous nonlinear function can be approximated arbitrarily well over a compact set by a multilayer ANN which consists of one or more hidden layers. ii) Learning and adaptation. The intelligence of ANNs comes from their generalization ability with respect to unknown data. On-line adaptation of the weights is possible. iii) Multivariable systems. ANNs may have many inputs and outputs, which makes it easy to model multivariable systems.

Manuscript received December 15, 2008. Several institutions contributed for this study: 1) Foundation of Science and Technology of Portugal, which financed the scholarship of investigation of doctorate SFR/16175/2004; 2) Laboratory for Process, Environmental and Energy Engineering (LEPAE), Department of Chemical Engineering, University of Porto; 3) Sugar refinery RAR, Portugal; 4) Company 30 de Novembro, Pinar del Río, Cuba; 5) Department of Telecommunications and Electronics, University of Pinar del Río, Cuba.

L.A.P.S. is with the Department of Chemical Engineering, Faculty of Engineering, University of Porto, R. Dr. Roberto Frias, 4200-465 Porto, Portugal (corresponding author, phone: +351-936706280; fax: +351-234370545; e-mail: lpaz@fe.up.pt).

P.G. is with the Department of Electronics Telecommunications and Informatics and with the Institute of Electrical Engineering and Telematics (IEETA), University of Aveiro, Portugal (e-mail: petia@ua.pt).

S.F.A. is with the Department of Chemical Engineering, Faculty of Engineering, University of Porto, Portugal (e-mail: sfeyo@fe.up.pt).

In the context of the present work we are mainly interested in studying the ability to project efficient ANN-based controllers for nonlinear systems with severe uncertainties and disturbances. This issue received an increasing attention [2] with ANNs being applied to design robust neural controllers with guaranteed stability and reference tracking. The neural control problem can be approached in direct or indirect control design framework. Direct ANN control means that the controller has an artificial neural network structure, while in the indirect ANN control scheme, first a ANN is used to model the process to be controlled, and this model is then employed in a more conventional controller design. The implementation of the first approach is simple but the design and the tuning are rather challenging. The indirect design is very flexible, the model is typically trained in advance and the controller is designed on-line, therefore it is the chosen scheme for the present work.

On other side, the concept of model-based predictive control (MPC), introduced in late seventies, nowadays has evolved to a mature level and became an attractive control strategy implemented in a variety of process industries [3]. The term MPC does not refer to a particular control method, instead it corresponds to a general control approach [4]. The main difference between MPC configurations is the model used to predict the future behavior of the process and the optimization procedure. First the MPC based on linear models gained popularity [5] as an industrial alternative to the classical proportional-integral-derivative (PID) control and later on nonlinear cases as polymerization reactors [6] and reactive distillation columns [7] were reported as successfully MPC controlled processes. The MPC controllers can cope with nonminimum phase and unstable processes, with nonlinearities and multi objectives. Further, they are easy to tune and process constrains can be handled systematically. All these features promote the practical applicability of predictive controllers as illustrated by a large number of reported applications.

The integration of MPC with ANN based models gained also an attention [8]-[9]. However the proposed solutions are computationally more involved, difficult to implement for processes with fast dynamics, and therefore the interest remained mainly at an academic level.

Based on the above considerations, the aim of this work is to study the computational viability of implementing the MPC control with ANN predictive process model to complex nonlinear systems. The proposed framework is illustrated for

a real industrial unstable in open loop process - a fed-batch sugar crystallization.

II. CLASSICAL VERSUS ERROR TOLERANT MPC

A. General MPC problem formulation

Nonlinear model predictive control (NMPC) is an optimisation-based multivariable constrained control technique that uses a nonlinear dynamic model for the prediction of the process outputs [4]. At each sampling time the model is updated on the basis of new measurements and state variables estimates. Then the open-loop optimal manipulated variable moves are computed over a finite (predefined) prediction horizon with respect to some performance index, and the manipulated variables for the subsequent prediction horizon are implemented. Then the prediction horizon is shifted or shrunk by usually one sampling time into the future, and the previous steps are repeated. The optimal control problem in the NMPC framework can be mathematically formulated as:

$$\min_{u_{\min} \leq u(t) \leq u_{\max}} J = \varphi(x(t), u(t), P), \quad (1)$$

subject to:

$$\dot{x} = f(x(t), u(t), P), \quad 0 \leq t \leq t_f, \quad x(0) = x_0 \quad (2.1)$$

$$y(t) = h(x(t), P) \quad (2.2)$$

$$g_j(x) = 0, \quad j = 1, 2, \dots, p \quad (3.1)$$

$$v_j(x) \leq 0, \quad j = 1, 2, \dots, l \quad (3.2)$$

where (1) is the performance index, (2) is the process model, function f is the state-space description, function h is the relationship between the output and the state, P is the vector of possibly uncertain parameters and t_f is the final time. $x(t) \in R^n$, $u(t) \in R^m$ and $y(t) \in R^p$ are the state, the manipulated input and the control output vectors, respectively. The manipulated inputs, the state and the control outputs are subject to the following constraints, $x(t) \in X$, $u(t) \in Z$, $y(t) \in Y$ in which X , Z and Y are convex and closed subsets of R^n , R^m and R^p . g_j and v_j are the equality and inequality constraints with p and l dimensions respectively.

B. Error tolerant digital nonlinear MPC control framework (main contribution)

Discrete process model. Considering the discrete nature of the on-line control problem the continuous time optimization problem involved in the NMPC formulation is solved by formulating a discrete approximation to it, that can be handled by conventional nonlinear programming (NLP) solvers. The time horizon $t = [t_0, t_f]$ is divided into equally spaced time intervals Δt , with discrete time steps

$$t_k = t_0 + k\Delta t \quad \text{and} \quad k = 0, 1, \dots, N.$$

The process model (2) is discretised as follows:

$$x(k+1) = F[x(k), u(k), P] \quad (4.1)$$

$$y_p(k) = h[x(k), P] \quad (4.2)$$

Process input constraints arise due to actuator limitations, such as saturation or rate-of-change restrictions, and can be expressed as

$$\begin{aligned} u_{\min} \leq u(t+k) \leq u_{\max}, \quad k = 1, 2, \dots, H_c \\ \Delta u_{\min} \leq \Delta u \leq \Delta u_{\max} \end{aligned} \quad (5)$$

where u_{\min} and u_{\max} are the minimum and the maximum values of the input, Δu_{\min} and Δu_{\max} are the minimum and the maximum values of the rate-of-change of the same input. *Process output constraints* are usually associated with operational limitations such as equipment specifications and safety considerations and can be expressed as:

$$y_{\min} \leq y_p(t+k) \leq y_{\max}, \quad k = 1, 2, \dots, H_p \quad (6)$$

We propose a modification of the general MPC formulation (1), where the optimization is performed based on the following performance index

$$u(t+k) = \begin{cases} \begin{cases} \min_{[u(t+k), u(t+k+1), \dots, u(t+H_c)]} J = \\ \lambda_1 \sum_{k=1}^{H_p} (e(t+k))^2 - \lambda_2 \sum_{k=1}^{H_c} (\Delta u)^2, & \text{if } |e(t+k)| > \alpha \\ u^* & \text{if } |e(t+k)| < \alpha \end{cases} \end{cases} \quad (7)$$

where $e(t+k) = r(t+k) - y_p(t+k)$,

$$\Delta u = u(t+k-1) - u(t+k-2)$$

subject to constraints (5) and (6).

(7) is a particular digital form of the general performance index defined by (1). We denote it as an *error tolerant MPC* formulation because the optimization is performed only when the error $e(t+k)$ is bigger than a predefined α value.

In order to reduce the computational burden when the absolute error $e(t+k)$ is smaller than α the control action is equal to u^* which is the last value of u , computed by the optimization procedure, before the error enters the α strip.

C. Selection of MPC parameters α , H_p , λ_2

α is a design parameter and its choice is decisive for achieving a reasonable compromise between lower

computational costs and still acceptable tracking error. While a formal analytical procedure for selecting α is still not found, the error tolerance is chosen based on common sense consideration of 5% to 10% around the set point. The prediction horizon H_p is the number of time steps over which the prediction errors are minimized and the control horizon H_c is the number of time steps over which the control increments are minimized, r is the desired response and y_p is the prediction model response. The choice of H_p is related with the sampling period (Δt) of the digital control implementation, which in its turn is a function of the settling time t_s (the time before entering into the 5% around the set-point) of the closed loop system. As a rule of thumb, it is suggested Δt to be chosen at least 10 times smaller than t_s , [10]. Hence, the prediction horizon can be chosen equal to $H_p = \text{round-to-integer}(t_s/\Delta t)$. It is well known that the smaller the sampling time, the better can a reference trajectory be tracked or a disturbance rejected. However, choosing a small sampling time yields a large prediction horizon. In order to compute the optimal control input, the optimization (1) is performed at each sampling time, therefore MPC controller requires a significant amount of on-line computation. Depending on the computer on which MPC is implemented, this can cause problems because of the large amount of memory and possibly causing numerical problems that are involved in using large prediction horizon. The introduction of the error tolerant MPC (7) serves as a compromise between these conflicting issues and reduces significantly the computational efforts.

$u(t+k), u(t+k+1), \dots, u(t+H_c)$ are tentative values of the future control signal, which are limited by u_{\min} and u_{\max} . Parameters λ_1 and λ_2 determine the contribution of the output error and the contribution of the control increments respectively on the performance index. In all control tests, summarized in section V, λ_1 was set to 1, as a normalized value for the importance of the first term. For the choice of λ_2 an empirical formula was deduced as follows,

$$(u_{\max} - u_{\min})^2 \cdot \lambda_2 = e_{p_{\max}} \cdot P/100 \quad (8)$$

where P defines the desired contribution of the second term in (7) ($0\% \leq P \leq 100\%$) and

$$e_{p_{\max}} = \max((r - y_{\max})^2, (r - y_{\min})^2) \quad (9)$$

The intuition behind (8-9) is to make the two terms of (7) compatible when they are not previously normalized and to overcome the problem of different numerical ranges of the two terms.

III. ARTIFICIAL NEURAL NETWORK (ANN) PROCESS MODELS

The ANNs appear to be rather convenient numerical models when dealing with strongly nonlinear systems or in general with systems for which data are available but little is known on the first principles mechanisms that define their dynamic behavior. The development of suitable algorithms for ANN training, as for example the Levenberg-Marquard (LM) algorithm, contributed for the increasing interest to the ANNs in the control community. The main advantages of the LM algorithm are in terms of execution time and robustness. However, the LM algorithm requires a lot of memory, therefore a powerful (in terms of memory) computer is the main condition for successful training. In order to solve problems of multiple local minima, typical for all derivative based optimization algorithms, the optimization is repeated several times specifying different starting points. The most popular ANN structures for modeling reasons are Feedforward networks (FFNN) and Recurrent (RNN) ones. Among various configurations, the RNN shown on Fig. 1 was selected as the most suitable for nonlinear system modeling.

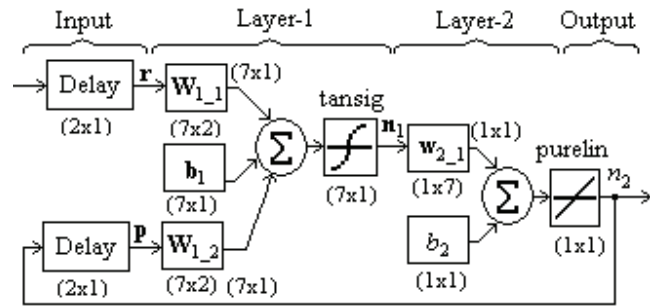


Fig. 1. RNN architecture

Tangent sigmoid hyperbolic activation functions are the hidden computational nodes (Layer 1) and a linear function is located at the output (Layer 2). Each NN has two vector inputs (r and p) formed by past values of the process input and the NN output respectively. The architecture of the ANN model is summarized as follows

$$u_{NN} = [r, p] \quad (10)$$

$$r = [u_c(t+k-1), u_c(t+k-2)] \quad (11)$$

$$p = [y_{NN}(t+k-1), y_{NN}(t+k-2)] \quad (12)$$

$$x = W_{11}r + W_{12}p + b_1 \quad (13)$$

$$n_1 = (e^x - e^{-x}) / (e^x + e^{-x}) \quad (14)$$

$$n_2 = w_{21}n_1 + b_2 \quad (15)$$

where $W_{11} \in R^{m \times 2}$, $W_{12} \in R^{m \times 2}$, $w_{21} \in R^{1 \times m}$, $b_1 \in R^{m \times 1}$, $b_2 \in R$ are the network weights (in matrix form) to be adjusted during the NN training, m is the number of nodes in the hidden layer. As mentioned above, the network training

is performed by LM algorithm, being one of the most celebrating quasi-Newton methods. The method is designed to approach second-order training speed without having to compute the Hessian matrix. Since the performance function has the form of a sum of error squares, the Hessian matrix can be approximated as

$$H(t+k) = J(t+k)^T J(t+k) \quad (16)$$

where $J(t+k)$ is the Jacobian matrix that contains first derivatives of the network errors ($e(t+k)$) with respect to the weights and biases

$$J(t+k) = \frac{\partial e(t+k)}{\partial w(t+k)}; \quad e(t+k) = r(t+k) - y_{NN}(t+k). \quad (17)$$

y_{NN} is the network output vector and r is the target vector. Then the gradient is computed as

$$g(t+k) = J(t+k)e(t+k), \quad (18)$$

and the LM algorithm updates the weights in the following way

$$w(t+k) = w(t+k) - [J(t+k)^T J(t+k) + \mu I]^{-1} g(t+k) \quad (19)$$

The NN model was trained with real industrial data. Different regression models were obtained based on data of two, four and six batches. The NN trained with six batches exhibits the best performance, therefore only results with this model are reported in section V. In order to extract the underlying nonlinear process dynamics a preprocessing of the initial industrial data was performed. From the complete time series corresponding to the input signal of one stage only the portion that really excites the process output of the same stage is extracted. Hence, long periods of constant (steady-state) behavior are discarded. Since, the steady-state periods for normal operation are usually preceded by transient intervals, the data base constructed consists (in average) of 60-70% of transient period data.

IV. INDUSTRIAL APPLICATION DESCRIPTION

Sugar crystallisation occurs through the mechanisms of nucleation, growth and agglomeration. There are two basic types of sugar production: from cane sugar or from beet. The process considered in this work is of the first type and a typical industrial unit can be divided into the following sequential phases [11].

Charging: During the first phase the pan is partially filled with a juice containing dissolved sucrose (termed liquor). The initial liquid charged in the pan corresponds approximately to 40% of the total vessel height. The charge

is usually performed by complete opening of the feeding valve until the level sensor indicates 40%. Therefore, no special control policy is required at this stage.

Concentration: The next phase is the concentration. The liquor is concentrated by evaporation, under vacuum, until the supersaturation reaches a predefined value. At this value seed crystals are introduced into the pan to start the production of crystals. This is the beginning of the crystallisation phase.

Crystallization (main phase): At this phase as evaporation takes place further liquor is added to the pan in order to guarantee crystal growth at a controlled supersaturation level and to increase the sugar content of the pan. Near to the end of this phase and for economical reasons, the liquor is replaced by other juice of lower purity (termed syrup).

Tightening: The fourth phase consists of tightening which is principally controlled by evaporation capacity. The pan is filled with a suspension of sugar crystals in heavy syrup, which is dropped into a storage mixer. At the end of the batch, the massecuite undergoes centrifugation, where final refined sugar is separated from (mother) liquor that is recycled to the process.

Sugar production is still a very heuristically operated process, with classical proportional integral and eventually derivative (PID) controllers being the most typical solution. However, the industrial partners ((Sugar Refinery RAR, Portugal, Company 30 de Noviembre, Pinar del Río, Cuba) agreed that an optimized operation policy might result in reduction of the recycled batches and thus in reduction of energy and material loss. These problems constitute the main motivation for the selection of the sugar crystallization as a real challenge in order to test the computational efficiency of the new ANN-MPC operation strategy. The second motivation is related with the fact that the crystallization phenomenon is typical for a great number of industrial processes such as for example in the pharmaceutical and food engineering. Therefore, the strategy formulated for this case study can be further extended and applied for other crystallization based processes.

The different phases of the sugar production are comparatively independent and moved by distinct driving forces, thus a single controller can hardly be effective for the complete process. Instead, individual ANN-MPC controllers for each stage where it seems appropriate, was the adopted framework (Fig. 2). See Table 1 for more details on the formulated operation strategy.

V. CONTROL TESTS

The most celebrating control algorithm in the industrial automation history is the Proportional Integral Derivative (PID). It is robust, comparatively simple to tune, with a lot of knowledge acquired with respect to the effect of each term. Therefore it is fare when introducing a new control strategy to compare it with the PID classical solution.

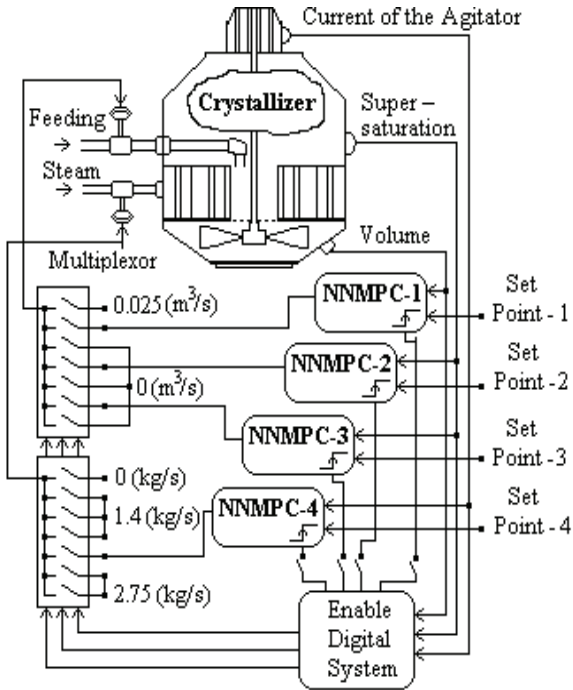


Fig.2 Cascade ANN-MPC control strategy

A. PID controllers

First, the PID parameters were tuned, where k_p , τ_i , τ_d are related with the general PID terminology as follows:

$$u(t+k) = k_p \left[e(t+k) + \frac{1}{\tau_i} \int_0^t e(t+k) dt + \tau_d \frac{de(t+k)}{dt} \right] \quad (20)$$

Since the process is nonlinear, classical (linear) tuning procedures were substituted by a numerical optimization of the integral (or sum in the digital version) of the absolute error (IAE):

$$IAE = \sum_{k=1}^N |r(t+k) - y_p(t+k)| \quad (21)$$

(21) was minimized in a closed loop framework between the digital process model and the PID controller. For each parameter an interval of possible values was defined based on empirical knowledge and the process operator expertise. A number of gradient (Newton-like) optimization methods were employed to compute the final values of each controllers summarized in Table 2. All methods concluded that the derivative part of the controller is not necessary. Thus, a PI controller was analyzed in the next tests.

B. Process performance results

The values of the MPC design parameters are summarized in Table 3. They are chosen based on the considerations discussed in section II. The operation strategy, summarized

in Table 1 and implemented by a sequence of ANN-MPC (Table 3) or PI controllers (Table 2) is comparatively tested in Matlab environment [12]. A process simulator was developed based on a detailed phenomenological model [13].

TABLE 2 OPTIMIZED PID PARAMETERS FOR THE CONTROL LOOPS DEFINED IN TABLE 1

	Control loop 1	Control loop 2	Control loop 3	Control loop 4
k_p	0.05	-0.5	20	-0.01
τ_i	30	40	10	70
τ_d	0	0	0	0

TABLE 3 MPC DESIGN PARAMETERS FOR THE CONTROL LOOPS DEFINED IN TABLE 1

control loop (CL)	t_s (s) settling time	Δt (s) sampl period	H_p predic. horizon	H_c control horizon	λ_2 weight	α error tolerance
CL1	40	4	10	2	1000	[12 12.3]
CL2	40	4	10	2	0.1	[1.142 1.158]
CL3	60	4	15	2	0.01	[1.142 1.158]
CL4	80	4	20	2	10000	[0.427 0.433]

A number of control tests were performed assuming different initial conditions, perturbations, sensor failure, etc. Due to space limits not all results are shown in this paper, moreover they all bring to the same conclusions summarized in Figs. 3-6, where the time trajectories of the controlled and the control (manipulated) variables of the four control loops are depicted. Both control strategies (PI and ANN-MPC) are equally successful in keeping the controlled outputs around their set points and in this sense the ANN-MPC does not bring significant tracking improvements. However, the main benefits of the ANN-MPC strategy are with respect to the batch end point performance. The results are summarized in Table 4, where the error tolerant ANN-MPC is compared not only with the PI strategy but also with ANN-MPC without an error tolerance (termed as classical ANN-MPC).

The quality of the produced crystals is evaluated by the particle size distribution (PSD) at the end of the process which is quantified by two parameters - the final average (in mass) particle size (MA), with reference 0.55 mm, and the final coefficient of particle variation (CV). Note that, the ANN-MPC controlled process ends up with much better AM and similar CV. All this is achieved with less energy, quantified by around 700 kilograms of vapor less consumed. Moreover, with less liquor consumption (about 1 m³) and shorter batch duration, the ANN-MPC controlled process produces the same quantity of sugar. In all tests, notable improvements of the final batch performance measures were registered when the proposed ANN-MPC control is applied.

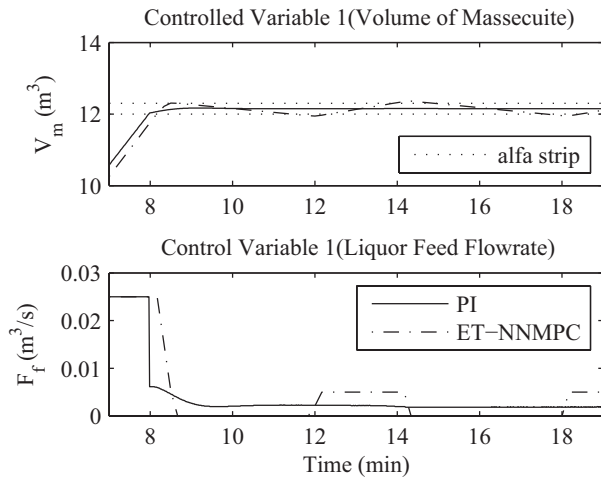


Fig. 3. Controlled (V_m – reactor volume) and control variables (F_f – feed flowrate) over time for the 1 control loop. Dashed line - error tolerant ANN-MPC; Full line – PI control

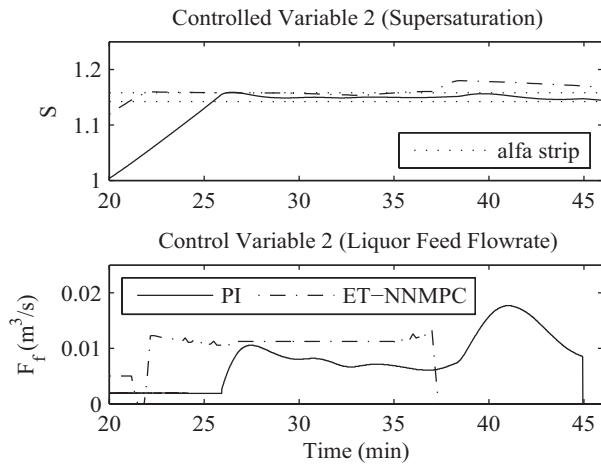


Fig. 4. Controlled (S – supersaturation) and the control variables (F_f – feed flowrate) over time for the 2 control loop. Dashed line - error tolerant ANN-MPC; Full line – PI control

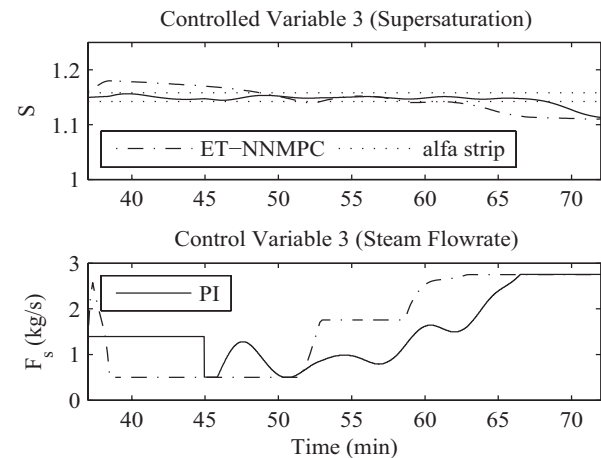


Fig. 5. Controlled (S – supersaturation) and the control variables (F_s – steam flowrate) over time for the 3 control loop. Dashed line - error tolerant ANN-MPC; Full line – PI control

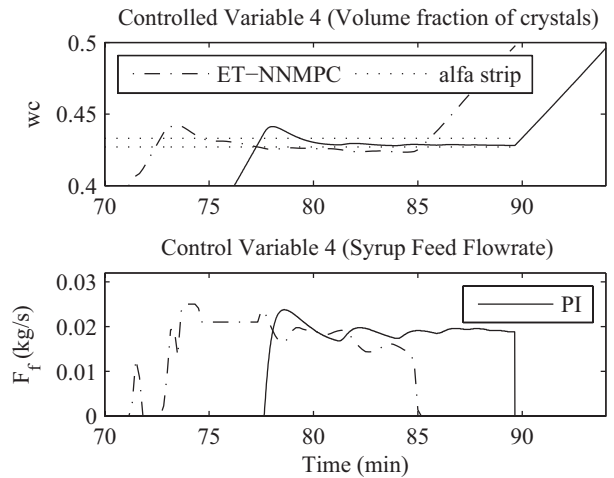


Fig. 6. Controlled (w_c – volume fraction of crystals) and the control variables (F_f – feed flowrate) over time for the 42 control loop. Dashed line - error tolerant ANN-MPC; Full line – PI control

TABLE 4 FINAL BATCH PERFORMANCE MEASURES

Performance measures	Error tolerant ANN-MPC (this paper)	Classical ANN-MPC	PI
AM (mm) (reference 0.55)	0.55	0.56	0.58
CV (%)	32	32	31
Batch duration (min:s)	89:40	89:00	94:15
Liquor consumption ,m ³	37.96	37.93	38.9
Energy consumption (Quantity of vapor in kg)	9204	9174	9907
Quantity of sugar production (t)	21.528	21.540	21.546

C. MPC computational time reduction

The intuition behind the idea of error tolerant MPC was to reduce the computational time and to make the ANN-MPC more attractive for real time implementation. In Fig. 7 are depicted the CPU times per iteration along the process duration for the classical MPC and the error tolerant MPC. The small error tolerances (see the last column of Table 3) brought a significant reduction of the computational time. The same analysis was performed between the error tolerant MPC and the classical MPC, with or without ANN-based process models and the average CPU times are summarized in Table 5. Though the results reported here are at an initial stage, the observed computational effort reduction of 10 times definitely deserves more attention. Further study is required to get a comprehensive and more systematic insight into the effect of the α parameter.

VI. CONCLUSIONS

The main contribution of this work is the introduction of a modified digital MPC control with ANN process model where the optimization is only performed when the tracking error is above a predefined level α . This modification reduces considerably the average duration of each

optimization step and makes the ANN-MPC algorithm computationally more efficient and attractive for industrial applications. To illustrate this statement the proposed strategy is implemented for the challenging nonlinear process of sugar production and compared with the traditionally applied PI controller and the classical MPC. An operation strategy with four control loops was formulated, with individual MPC or PI controllers for each process stage. The results have shown that the error tolerant ANN-MPC and the PI controllers perform equally good with respect to the individual loop reference tracking. However, the ANN-MPC clearly outperforms the PI strategy with respect to the final batch performance measures (less energy consumption, improved crystal size parameters, higher productivity and shorter batch time).

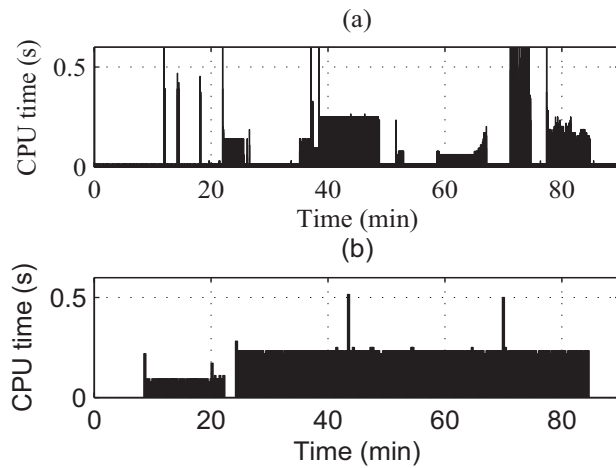


Fig. 7 CPU time per iteration along the process duration. (a)-Error tolerant ANN-MPC – eq. (7); (b) Classical MPC

TABLE 5 AVERAGE CPU TIME

MPC control	Average CPU time (s)
Classical MPC	0.17
Error tolerant MPC	0.099
ANN-MPC	0.12
Error tolerant ANN-MPC (this paper)	0.08

REFERENCES

- [1] S. Haykin, (1999) *Neural Networks: A Comprehensive Foundation*, Prentice Hall, NJ.
- [2] M. Norgaard, O. Ravn, N. K. Poulsen and L. K. Hansen (2000), *Neural networks for modelling and control of dynamic systems* -. Springer-Verlag, London.
- [3] E. F. Camacho and C. Bordodns (2004) *Model Predictive Control in the process industry*, London, Springer-Verlag.
- [4] J. A. Rossiter (2003) *Model based predictive control. A practical approach*, New York, CRC Press.
- [5] Morari, M. (1994). *Advances in Model-Based Predictive Control*. Oxford: Oxford University Press.
- [6] H. Seki, M. Ogawa, S. Ooyama, K. Akamatsu, M. Ohshima and W. Yang (2001) Industrial application of a nonlinear model predictive control to polymerization reactors, *Control Engineering Practice*, 9, 819-828
- [7] L. S. Balasubramhanya and F. J. Doyle, (2000) Nonlinear model-based control of a batch reactive distillation column. *Journal of Process Control*, 10, 209-218.
- [8] D. W. Yu1 and D. L. Yu, Neural network control of multivariable processes with a fast optimisation algorithm, *Neural Computing & Applications*, 12, 185-189, 2003.
- [9] Yu-Jia Zhai and Ding-Li Yu, A Neural Network Model Based MPC of Engine AFR with Single-Dimensional Optimization, *Lecture Notes in Computer Science*, D. Liu et al. (Eds.), Part I, LNCS 4491, pp. 339–348, 2007.
- [10] Soeterboek, R. (1992). *Predictive control. A unified approach*. Prentice Hall International Series in Systems and Control Engineering.
- [11] V. Galvanauskas, P. Georgieva, and S. F. de Azevedo. (2006) Dynamic Optimisation of Industrial Sugar Crystallization Process based on a Hybrid (mechanistic+ANN) Model. *IEEE World Congress on Computational Intelligence*. 16-21 July 2006, Vancouver, Canada.
- [12] A. Bemporad, M. Morari and N. L. Ricker , (2005) *User's Guide: Model Predictive Control Toolbox for use with MatLab, Version 2*. The MathWorks Inc.
- [13] P. Georgieva,, M. J . Meireles and S. F. de Azevedo (2003) Knowledge Based Hybrid Modeling of a Batch Crystallization When Accounting for Nucleation, Growth and Agglomeration Phenomena. *Chemical Engineering Science*, 58, 3699-3707.

TABLE 1- SUMMARY OF THE OPERATION STRATEGY FOR THE EVAPORATIVE FED-BATCH SUGAR CRYSTALLIZATION PROCESS

Stage	Action	Control
Charge	<ol style="list-style-type: none"> 1. The steam valve is closed. 2. The stirrer is off. 3. The vacuum pressure changes from 1 to 0.23 <i>bar</i>. 4. The vacuum pressure reaches 0.5 <i>bar</i>, feeding starts with max rate. 5. Liquor covers 40 % of the vessel height. 	No control The feed valve is completely open
Concentration	<ol style="list-style-type: none"> 1. The vacuum pressure stabilizes around 0.23 <i>bar</i>. 2. The stirrer is on. 3. The volume is kept constant . 4. The steam flowrate increases to 2 <i>kg/s</i> 5. The supersaturation reaches 1.06, the feeding is closed, the steam flowrate is reduced to 1.4 <i>kg/s</i> 	<i>Control loop 1</i> Controlled variable: Volume; Manipulated variable: liquor feed flowrate
Seeding and setting the grain	<ol style="list-style-type: none"> 1. The supersaturation reaches 1.11. 2. Seed crystals are introduced. 3. The steam flowrate is kept at the minimum for two minutes. 	No control The feed valve is closed
Crystallization (phase 1)	<ol style="list-style-type: none"> 1. The steam flowrate is kept around 1.4 <i>kg/s</i>. 2. The supersaturation is controlled at the set point 1.15. 	<i>Control loop 2</i> Controlled variable: supersaturation Manipulated variable: liquor feed flowrate
Crystallization with liquor (phase 2)	<ol style="list-style-type: none"> 1. The volume of crystallizer reaches $\approx 22 \text{ m}^3$. 2. The feed valve is closed. 3. The supersaturation is controlled at the set point 1.15. 4. The stirrer power reaches 20.5 <i>A</i>. 	<i>Control loop 3</i> Controlled variable: supersaturation Manipulated variable: steam flowrate
Crystallization with syrup	<ol style="list-style-type: none"> 1. The steam flowrate is kept around the maximum of 2.75 <i>kg/s</i>. (<i>hard constrain</i>). 2. The volume fraction of crystals is kept at the set point 0.45. 3. The volume reaches its maximum value (30 m^3) 4. The feed valve is close. 	<i>Control loop 4</i> Controlled variable: volume fraction of crystals. Manipulated variable: syrup feed flowrate
Tightening	<ol style="list-style-type: none"> 1. The stirrer power reaches the maximum value of 50 <i>A</i> (<i>hard constrain</i>). 2. The steam valve is closed. 3. The stirrer and the barometric condenser are stopped. 	No control