2014 European Control Conference (ECC)
June 24-27, 2014. Strasbourg, France

Evaluation of the NEPSAC Nonlinear Predictive Controller on a Thermal Process

Robin De Keyser and Andres Hernandez

Abstract— **Nonlinear dynamics are commonly encountered in industrial applications, where manufacturing of higher quality products very often requires that the process works within a wide range of operating conditions close to the boundaries. Nonlinear Model Predictive Control (NMPC) appears as a solution due to its capability to find optimal control actions for the case of nonlinear processes with constraints. In this contribution, the control problem is solved using the Nonlinear Extended Prediction Self-Adaptive Control (NEPSAC) approach to model predictive control (MPC), which besides of being a fast algorithm also avoids explicit local linearization by directly using the nonlinear model for prediction. The effectiveness of the mentioned nonlinear controller and the procedure to express a nonlinear model suitable for prediction is illustrated on a simulation example of a highly nonlinear thermal process. Furthermore, the benefits of NEPSAC are clearly shown by comparing its performance to linear controllers such as linear MPC, PI and PID controllers.**

I. INTRODUCTION

Model Predictive Control (MPC) refers to a family of control approaches, which makes explicit use of a model of the process to optimally obtain the control signal by minimizing an objective function [1]. The effectiveness of MPC strategies has been recognized due to the multiple successful implementations in real-life applications as discussed in [2], [3]. However, most of the implemented MPC techniques are based on a linear model, although real processes are in general inherently nonlinear. This, together with higher product quality specifications and increasing productivity demands, tighter environmental regulations and demanding economical considerations in the process industry require to operate systems closer to the boundary of the admissible operating region, endangering the performance of linear control techniques.

Linear predictive control is predominantly used because it is easier and faster to obtain a linear model compared to a nonlinear one. Another factor is that stability and robustness are more difficult to address in the case of a nonlinear MPC (NMPC). Additionally, some of the nonlinear models and/or constraints lead to nonconvex nonlinear optimization problems that are relatively complex to solve. Due to these factors, the application of the nonlinear MPC in practical situations is still very limited although its potential is promising [1].

Andres Hernandez acknowledge the financial support provided by the Institute for the Promotion and Innovation by Science and Technology in Flanders (IWT SBO-110006).

R. De Keyser and A. Hernandez are with Ghent University, Faculty of Engineering and Architecture, Department of Electrical energy, Systems and Automation, Sint Pietersnieuwstraat 41, 9000 Gent, Belgium. e-mail: {Robain.DeKeyser; Andres.Hernandez}@UGent.be

It is important to consider that in the case of a highly nonlinear system, a single linear model cannot provide acceptable results in all operating regions. In other words, it cannot be linearly modeled to be adequate in all operating regions, unless the process always works around the operating point. A popular approach to approximately model highly nonlinear systems includes the use of multiple linear models in different operating points. Subsequently, several local linear controllers can be 'scheduled' to yield acceptable control system performance [4], [5]. However, in many situations, coming up with a satisfactory controller schedule can consume far more human resources than the linearization and linear control design tasks. Moreover, the stability of the resulting system cannot be guaranteed. In [6] it is proposed another approach of model predictive control for nonlinear systems, in which the model is adaptively linearized along the prediction horizon in order to improve the performance of the controller, especially for the case of large prediction horizons. In that approach, the optimum results of the previous sample time are utilized for linearization at the current sample time, thus leading to a better control performance.

In this contribution, the benefits of the Nonlinear Extended Prediction Self-Adaptive Control (NEPSAC) approach to NMPC are emphasized. This controller, fully described in [7], uses the nonlinear model for prediction directly, taking full advantage of the given nonlinear system dynamics to generate a high-performance design without involving model linearization or gain scheduling for its implementation. Additionally, the usually complex nonlinear optimization problem is in the NEPSAC algorithm replaced by a more simple iterative quadratic programming procedure. Furthermore, the NEPSAC performance is evaluated against several linear controllers including the linear MPC and the classical PI and PID controllers.

The content of this paper is as follows. The process description and modeling is presented in section II. Next the Model Predictive Control methodology is briefly introduced in section III. The performance of the proposed nonlinear controller NEPSAC is then compared against linear model predictive control, PI and PID in section IV. Finally a conclusion section summarizes the main outcome of this investigation.

II. PROCESS DESCRIPTION

The process considered in this paper consists of a heated tank of which the level is controlled by a mechanical float switch, resulting in a constant water volume V (Fig. 1). A

submerged electrical heater delivers a *constant* heat flow Q, which causes the liquid to warm up. Temperature control of the outlet water is achieved by changing the outflow $q(t)$ of hot tank water, which allows an equal amount of cold tap water to flow in.

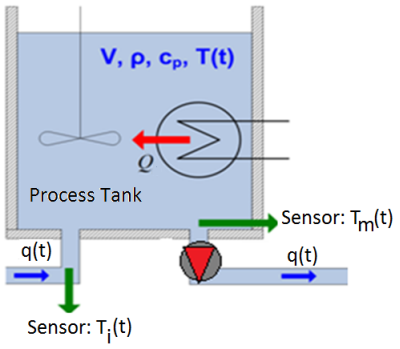

Fig. 1. Schematic representation of the process.

Accurate flow control is achieved using a peristaltic pump driven by a 24 V DC-motor, that provides linearity between control voltage and flow. Finally, two PT100 sensors measure the temperature: (1) of the cold incoming tap water $T_i(t)$ and (2) of the tank $T(t)$.

A. Process modelling

The mathematical model that describes the relationship between the outflow $q(t)$ and the tank temperature $T(t)$ follows from the energy balance equation:

$$\rho\, c_p\, V \frac{dT(t)}{dt} = Q + \rho\, c_p\, q(t)\,(T_i(t) - T(t)) \tag{1}$$

where $\rho = 1\,\mathrm{Kg}/l$ and $c_p = 4186\,\mathrm{J/Kg^{\circ}C}$ are respectively the density and the specific heat of water, $V = 1\,l$ is the volume in the tank and $Q = 6000\,\mathrm{W}$ is the constant amount of supplied heat. The small heat losses to the environment are negligible. Notice that the model is nonlinear, as $q(t)$ is the control input (manipulated variable). The flow range is $0 \le q \le 0.1\ l/\mathrm{s}$, or equivalently, $0 \le q \le 6\ l/\mathrm{min}$.

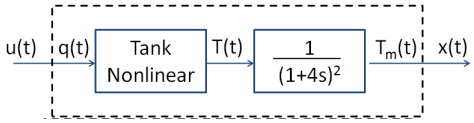

Fig. 2. Input and output scheme.

To account for the dynamics of the actuator and sensor, a second order transfer function (2) is considered as depicted in Fig. 2. Notice that $T(t)$ becomes a virtual variable representing the instantaneous change of temperature in the tank used just for modelling purposes, while $T_m(t)$ corresponds to the actual measured value.

$$\frac{T_m(s)}{T(s)} = \frac{1}{(1+4s)^2} \tag{2}$$

This kind of system is of interest for control engineers because in many industrial applications the dynamics of the system changes as a function of the manipulated variable (e.g. heat exchangers). For this particular thermal process, both the gain and time constant change as a function of the flow $q(t)$. This can be analised from (1) but also graphically observed by implementing a staircase experiment, as shown in Fig. 3. Although the steps applied to the flow are of the same magnitude, these have a different effect on the temperatute depending on the current state of the system. This is, nonetheless, not unexpected as this is the main characteristic of any nonlinear system. For instance, during the step applied from 200 to 400 seconds, the temperature decreases about $20^{\circ}C$ in 200 s while the temperature decreases just $5^{\circ}C$ in about 50 s during the step applied from 1200 to 1400 seconds.

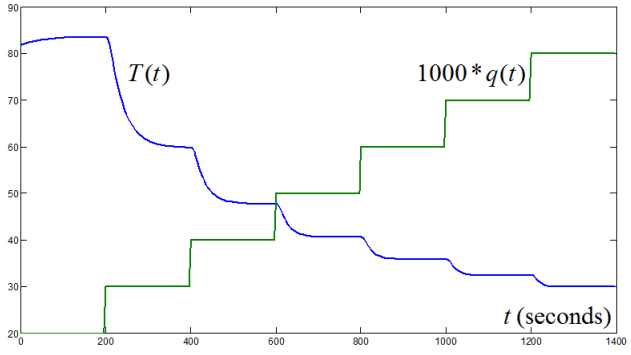

Fig. 3. Staircase experiment over the full range of the system.

B. Linearization of the model

In order to design the linear controllers (MPC, PID) which will be used in the comparative study in section IV, the nonlinear model (1) is linearized around the operating point $(\overline{q}, \overline{T})$ and represented in the Laplace operator 's' using deviation values:

$$\frac{\triangle T(s)}{\triangle q(s)} = \frac{K}{1+\tau s} \tag{3}$$

with $\tau = \frac{V}{\overline{q}}$ and $K = -\frac{Q}{\rho C_p \overline{q}^2}$.

The transfer function (3) is evaluated in 4 different points to illustrate the important change in both time constant τ and gain K as summarized in table I.

TABLE I
LINEARIZED MODEL AT 4 DIFFERENT POINTS

$\overline{T}$ $^{\circ}C$	$\overline{q}$ l/s	$\overline{q}$ l/min	K $^{\circ}Cs/l$	τ s
30	0.0796	4.78	-226	13
60	0.0299	1.79	-1607	33
90	0.0184	1.10	-4245	54
50	0.0377	2.26	-1007	27

By considering the steady-state values of $T(t)$ in Fig. 3 and plotting them against the corresponding value of the flow $q(t)$, the static characteristic of the plant is built. This relationship describes the nonlinearity of the system as depicted in Fig. 4. In this study, the temperature of $\overline{T} = 50^{\circ}C$ in the middle of the plant's range is chosen as a working point. It should be noted that the modelling error increases

nonlinearly by deviating from the linear model represented with the dotted red line.

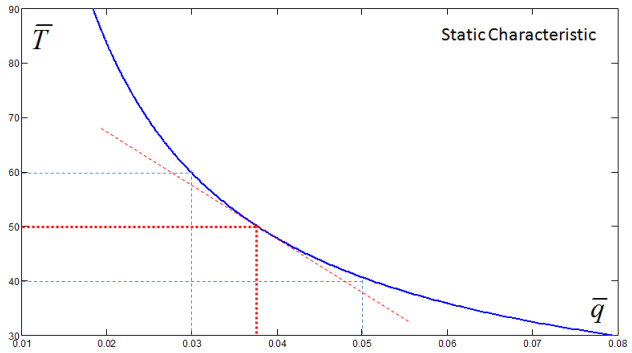

Fig. 4. Static characteristic of the system over the full operation range.

Eventually, the transfer function of the process including the sensor and actuator dynamics at $\overline{T} = 50°C$ is obtained:

$$\frac{\triangle T_m(s)}{\triangle q(s)} = \frac{-1007}{(1+27s)(1+4s)^2} \tag{4}$$

III. MODEL PREDICTIVE CONTROL

Among the different MPC methodologies available in literature, the Extended Prediction Self-Adaptive Control (EPSAC) proposed by [7] has been chosen as it presents interesting features, especially in its nonlinear version (NEPSAC). A brief introduction to this algorithm is presented in this section.

A. EPSAC algorithm

In the EPSAC algorithm, the model output $x(t)$ represents the effect of the control input $u(t)$ on the process output $y(t)$. It can be described by the following equation:

$$x(t) = f[x(t-1), x(t-2), \ldots, u(t-1), u(t-2), \ldots] \tag{5}$$

Notice that $x(t)$ represents here the model output, not the state vector. Also important is the fact that f can be either a linear or a nonlinear function. The generic model of the EPSAC algorithm is:

$$y(t) = x(t) + n(t) \tag{6}$$

where $y(t)$ is the measured output of the process, $x(t)$ is the model output and $n(t)$ represents model/process disturbance, all at discrete-time index t. The disturbance $n(t)$ can be modeled as coloured noise through a filter with the transfer function:

$$n(t) = \frac{C(q^{-1})}{D(q^{-1})} e(t) \tag{7}$$

with $e(t)$ uncorrelated (white) noise with zero-mean and C, D monic polynomials in the backward shift operator q^{-1}. The disturbance model must be designed to achieve robustness of the control loop against unmeasured disturbances and modelling errors. A 'default' choice to remove steady-state control offsets is $n(t) = \frac{1}{1-q^{-1}} e(t)$ [8].

A fundamental step in the MPC methodology consists of the prediction. Using the generic process model (6), the predicted values of the output are:

$$y(t+k|t) = x(t+k|t) + n(t+k|t) \tag{8}$$

for $k = N_1 \ldots N_2$, where N_1 and N_2 are the minimum and the maximum prediction horizons. The prediction of the process output is based on the measurements available at sampling time instant t, $\{y(t), y(t-1), \ldots, u(t-1), u(t-2), \ldots\}$ and future (postulated) values of the input signal $\{u(t|t), u(t+1|t), \ldots\}$. The future response can then be expressed as:

$$y(t+k|t) = y_{base}(t+k|t) + y_{opt}(t+k|t) \tag{9}$$

The two contributing factors have the following origin:

- $y_{base}(t+k|t)$ is the effect of the past inputs $u(t-1), u(t-2) \ldots$, a future base control sequence $u_{base}(t+k|t)$ (which is pre-specified, ref. section III-B) and the predicted disturbance $n(t+k|t)$.
- $y_{opt}(t+k|t)$ is the effect of the optimizing control actions $\delta u(t|t), \ldots, \delta u(t+N_u-1|t)$ with $\delta u(t+k|t) = u(t+k|t) - u_{base}(t+k|t)$, in a control horizon N_u.

The optimized output can be expressed as the discrete-time convolution of the unit impulse response coefficients $h_1, \ldots, h_{N_2}$ and unit step response coefficients $g_1, \ldots, g_{N_2}$ of the system as follows:

$$\begin{aligned} y_{opt}(t+k|t) = h_k \delta u(t|t) + h_{k-1} \delta u(t+1|t) + \ldots \\ + g_{k-N_u+1} \delta u(t+N_u-1|t) \end{aligned} \tag{10}$$

Using (9) and (10), the key EPSAC-MPC formulation becomes:

$$\mathbf{Y} = \overline{\mathbf{Y}} + \mathbf{G}.\mathbf{U} \tag{11}$$

where:

$$\begin{aligned} \mathbf{Y} &= [y(t+N_1|t) \ldots y(t+N_2|t)]^T \\ \overline{\mathbf{Y}} &= [y_{base}(t+N_1|t) \ldots y_{base}(t+N_2|t)]^T \\ \mathbf{U} &= [\delta u(t|t) \ldots \delta u(t+N_u-1|t)]^T \end{aligned} \tag{12}$$

$$\mathbf{G} = \begin{bmatrix} h_{N_1} & h_{N_1-1} & \ldots & g_{N_1-N_u+1} \\ h_{N_1+1} & h_{N_1} & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots \\ h_{N_2} & h_{N_2-1} & \ldots & g_{N_2-N_u+1} \end{bmatrix} \tag{13}$$

Once the output is predicted, it is possible to optimize the control signal $\mathbf{U}$ by minimizing the cost function:

$$J(\mathbf{U}) = \sum_{k=N_1}^{N_2} [r(t+k|t) - y(t+k|t)]^2 \tag{14}$$

Notice that the controller cost function (14) can be easily extended to many alternative cost functions (similar to the approach in optimal control theory) as described in [7]. The *horizons* N_1, N_2 are design parameters and $r(t+k|t)$ is the desired *reference trajectory*, chosen here as a 1^{st}-order reference trajectory as specific implementation example:

$$r(t+k|t) = \alpha r(t+k-1|t) + (1-\alpha) w(t+k|t) \tag{15}$$

for $k = 1 \ldots N_2$ and initialization $r(t|t) = y(t)$. The signal $w(t)$ represents the setpoint and α a design parameter that plays an important role in tuning the MPC performance [9]. The optimal input solution of the EPSAC algorithm can be written in matrix form:

$$\mathbf{U}^* = [\mathbf{G}^\mathbf{T}\mathbf{G}]^{-\mathbf{1}} [\mathbf{G}^\mathbf{T}(\mathbf{R} - \overline{\mathbf{Y}})] \tag{16}$$

with R being the vector notation of the reference trajectory, $\mathbf{R} = [r(t+N_1|t) \ldots r(t+N_2|t)]^T$ and $[G^T G]$ of dimension $N_u x N_u$. Only the first optimal control input $u(t) = u_{base}(t/t) + \delta u(t/t)$ is applied to the plant and the whole procedure is repeated again at the next sampling instant $t+1$.

B. NEPSAC algorithm

The calculation of the predicted output with (9) involves the superposition principle. When a nonlinear system model $f[.]$ is used in (5), above strategy is only valid -from a practical point of view - if the term $y_{opt}(t+k|t)$ in (9) is for example 10 times smaller than the term $y_{base}(t+k|t)$. When this term would be zero, the superposition principle would no longer be involved. The term $y_{opt}(t+k|t)$ will be small if $\delta u(t+k|t)$ is small, see (10).

This can be realized iteratively, by executing the following steps at each controller sampling instant:

1. Initialize $u_{base}(t+k|t)$ as: $u^1_{base}(t+k|t) = u^*(t+k|t-1)$, i.e. the optimal control sequence as computed during the previous sampling instant; in other words: $u^*(t+k|t-1)$ is used as a first estimate for $u^*(t+k|t)$.
2. Compute the G matrix using directly the nonlinear model of the process (5) to obtain (13).
3. Calculate $\delta u^1(t+k|t)$ using the linear EPSAC algorithm.
4. Calculate the corresponding $y^1_{opt}(t+k|t)$ with (10) and compare it to $y^1_{base}(t+k|t)$, which is the result of $u^1_{base}(t+k|t)$.
5. In case $y^1_{opt}(t+k|t)$ is *not* small enough compared to $y^1_{base}(t+k|t)$: re-define $u_{base}(t+k|t)$ as $u^2_{base}(t+k|t) = u^1_{base}(t+k|t) + \delta u^1(t+k|t)$ and go to 2. The underlying idea is that $u^1_{base}(t+k|t) + \delta u^1(t+k|t)$ - which is the optimal $u^*(t+k|t)$ for a linear system - can act as a second estimate for the optimal $u^*(t+k|t)$ in case of a nonlinear system.
6. In case $y^i_{opt}(t+k|t)$ is small enough compared to $y^i_{base}(t+k|t)$: use $u(t) = u^i_{base}(t+k|t) + \delta u^i(t+k|t)$ as the resulting control action of the current sampling instant (notice that $i = 1,2,\ldots,$) according to the number of iterations)

This algorithm results after convergence to the optimal solution for the underlying nonlinear predictive control problem. The number of required iterations depends on how far the optimal $u^*(t+k|t)$ is away with respect to $u^*(t+k|t-1)$. In quasi-steady-state situations, the number of iterations is low (1 ... 2). On the other hand, during transients the number of iterations might raise to 10 (refer to Fig. 9).

IV. CONTROL PERFORMANCE

In this section, the performance of the NEPSAC is evaluated, using as a reference the classical PI/PID controller on the one hand, and the linear MPC on the other hand. Notice that in all circumstances, the incoming cold water T_i and the heater power Q are considered to be constant, equal to $12\,^{\circ}C$ and 6000 W respectively.

A. PI and PID

A PI controller has been designed to compensate the slowest time constant of the system (4), giving as result the parameters $K_p = -0.001$ and $T_i = 27$. The performance of the controller is tested with and without implementing an Anti Reset-Windup (ARW) as depicted in Fig. 5. It is noticeable how the performance of the controller with ARW improves, by decreasing the big overshoot presented during the start-up. Although this example clearly shows the benefit of introducing ARW schemes, without the need of re-designing the controller, its implementation is sometimes forgotten in practice.

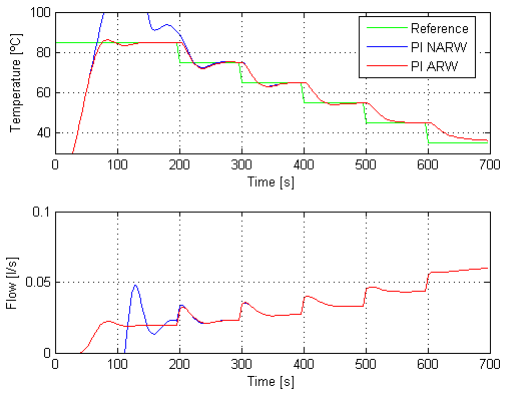

Fig. 5. Comparison PI controller with and without anti reset-windup (ARW).

The PID controller was tuned using the Frequency Response toolbox (FRtool) described in [10], to compensate the process time constants 27 and 4, thus resulting in the controller parameters: $K_p = -0.002$, $T_i = 31$ and $T_d = 3.5$. The performance of the PID controller is tested also in both cases with and without anti reset-windup (ARW) as presented in Fig. 6.

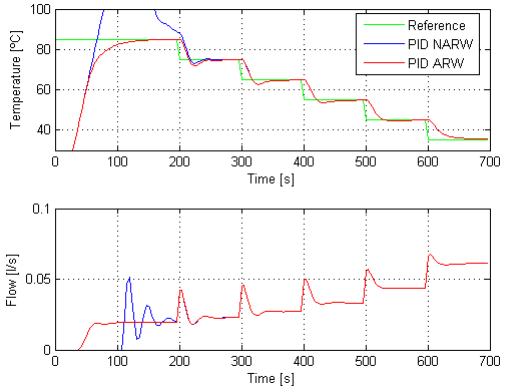

Fig. 6. Performance comparison between PID with and without anti reset-windup (ARW).

The PID is faster compared to the PI as observed in Fig. 6 during the period from 500 to 600 seconds. Nevertheless, this is another aspect which is very often neglected in practice as tuning the derivative action is not an obvious task.

B. Linear Model Predictive Control (EPSAC)

The MPC involves the use of a model suitable for prediction. In the case of the EPSAC algorithm, we require to express the output $T_m(t)$ as a function of the previous measured values as in (5). For the linear case, this can be obtained in a straightforward manner by discretizing (4) using a sampling time of $T_s = 4\,$s. Using then the equivalent

notation presented in Fig. 2 for the *input* $u(t) \equiv q(t)$ and for the *output* $x(t) \equiv T_m(t)$, it results in:

$$x(t) = 1.598x(t-1) - 0.7698x(t-2) + 0.1167x(t-3) \\ - 14.85u(t-1) - 35.48u(t-2) - 5.073u(t-3) \quad (17)$$

The performance of the EPSAC controller tuned using $N_u = 1$, $N_1 = 1$, $N_2 = 5$ and $\alpha = 0.5$ is depicted in Fig. 7.

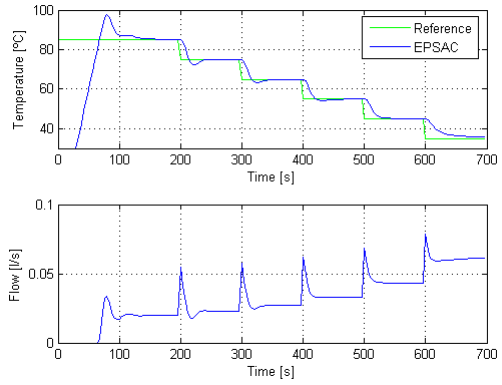

Fig. 7. Linear Model Predictive Control (EPSAC).

C. Nonlinear Model Predictive Control (NEPSAC)

As for the linear case, NEPSAC requires a discrete-time model of the system suitable for prediction. The procedure to express the nonlinear model as in (5), starts by dicretizing the nonlinear differential equation (1), resulting in:

$$T(t) = T(t-1) + \frac{T_s}{V}\left[\frac{Q}{\rho C_p} + q(t-1)\left[T_i - T(t-1)\right]\right] \quad (18)$$

The two first-order filters representing sensor and actuator (2) are replaced by their discrete-time representation again using the equivalent notation for *input* $u(t) \equiv q(t)$ and *output* $x(t) \equiv T_m(t)$:

$$x(t) = 0.736\ x(t-1) - 0.135\ x(t-2) + 0.399\ T(t) \quad (19)$$

Representing $T(t-1)$ from (19), it follows that:

$$T(t-1) = \frac{1}{0.399}\left[x(t-1) - 0.736x(t-2) + 0.135x(t-3)\right] \quad (20)$$

The procedure to calculate $x(t)$ at time t from previous measured values is summarized as follows:

1. Calculate $T(t-1)$ from (20) using $[x(t-1)\ldots]$
2. Calculate $T(t)$ from (18) using $u(t-1)$ or $\equiv q(t-1)$
3. Calculate $x(t)$ from (19)

Finally, an excellent performance of the controller was achieved for the design parameters: $N_u = 1$, $N_1 = 1$, $N_2 = 5$, $\alpha = 0.5$ as depicted in Fig. 8.

An important aspect to be discussed regarding the NEPSAC implementation is the computation of the G matrix, as mentioned in section III-B. One of the advantages of the NEPSAC methodology is the fact that it does not require of explicit linearization of the nonlinear model, instead it computes the G matrix from the coefficients of the step applied to the nonlinear model (5). An *alternative* procedure consists in computing the G matrix from the step response coefficients of the linearized model of the complete system (4), leading however to a lower closed-loop performance especially for the case of large changes in the setpoint.

The results obtained for both the correct and alternative implementations of NEPSAC are depicted in Fig. 8, where *LG* and *NLG* represent the linearized and nonlinearized approaches to compute the G matrix, respectively. There are quite large differences between the two implementations, as the control effort for the linearized implementation is higher it results in overshoot for big changes in the setpoint and a higher number of iterations as depicted in Fig. 9. Notice that as expected the number of iterations for both cases increases at the moment that a change in the setpoint occurs, and remains in one during steady-state.

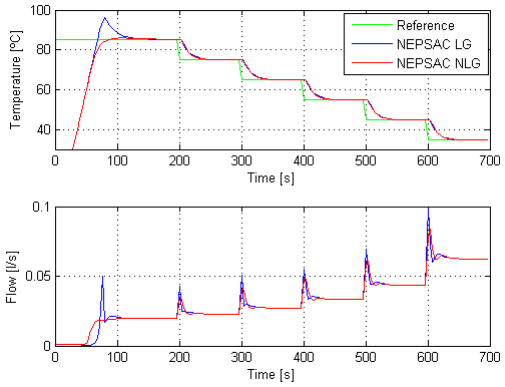

Fig. 8. Nonlinear Model Predictive Control (NEPSAC) performance for two different implementations using a linearized *LG* and nonlinearized *NLG* way of computing the G matrix.

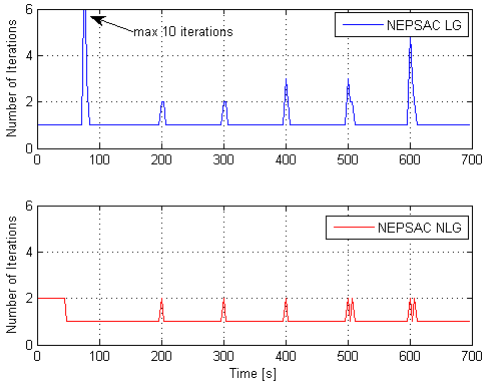

Fig. 9. Number of iterations required for NEPSAC for the two different implementations using a linearized *LG* and nonlinearized *NLG* way of computing the G matrix.

Comparison details at *Design* temperature $\overline{T} = 50°C$

Special attention is paid now to the performance of the controllers close to the design temperature $\overline{T} = 50°$C. All controllers perform similar except the PI controller as observed in Fig. 10. As mentioned above in section IV-A, this is the disadvantage of excluding the derivative action. A less agressive control effort is achieved with NEPSAC NLG compared to the NEPSAC LG and EPSAC controllers.

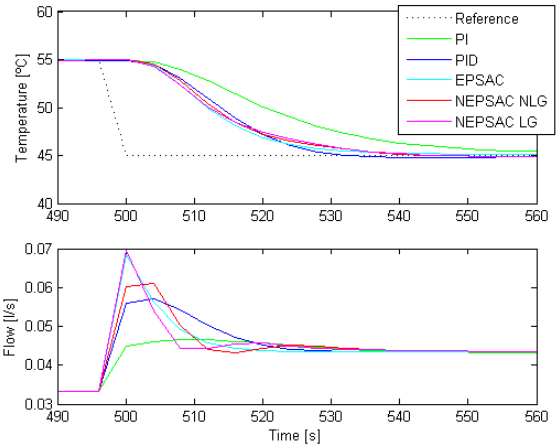

Fig. 10. Comparison at design temperature $\overline{T} = 50°C$

Comparison details at *Low* temperature $\overline{T} = 40°C$

At low temperature the PI again responds worst, while the PID and EPSAC controllers achieve similar results in terms of settling time and overshoot despite the slight difference in the control effort as shown in Fig. 11. In this temperature range the NEPSAC outperforms the other controllers although it requires a bigger control effort. Also in this case the differences between the correct and alternative ways to compute the G matrix in NEPSAC are visible, where the correct implementation EPSAC NLG describes a smoother control effort.

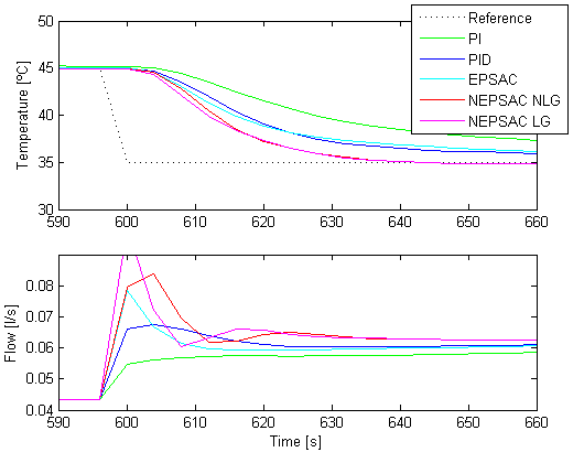

Fig. 11. Comparison at low temperature $\overline{T} = 40°C$.

Comparison details at *High* temperature $\overline{T} = 80°C$

At high temperature, the difference in the performance becomes more evident, as illustrated in Fig. 12. The PID and EPSAC both react similarly aggressive, also presenting an overshoot of about 30%. The PI presents the worst performance as it is a sluggish controller, and therefore slow, presenting the same overshoot as the other linear controllers. Finally, the NEPSAC controller produces a desired performance due to its fast response without overshoot and smallest control effort. Although, in this temperature range the NEPSAC NLG and NEPSAC LG have similar performance, still NEPSAC NLG presents a preferred, less aggressive, control action.

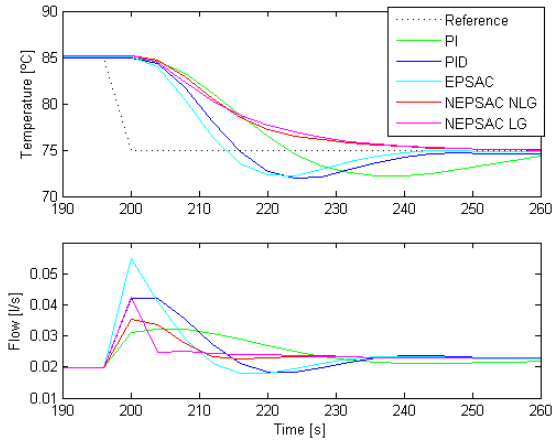

Fig. 12. Comparison at high temperature $\overline{T} = 80°C$.

V. CONCLUSIONS

In the present contribution, an effective and efficient nonlinear predictive controller has been evaluated and compared against linear controllers. First, a complete methodology to represent a suitable model for prediction has been presented for linear and nonlinear predictive control. Second, it has been illustrated by means of a simulation example the great benefit of NEPSAC to deal with processes that require a wide operation range. Finally, insight in the computation of the G matrix for nonlinear systems is described, and compared to the performance obtained by linearizing the system each sample around the current operating point.

REFERENCES

[1] E.F. Camacho and C., Bordons, (2007) Model Predictive Control. ISBN-13: 978-1852336943, *Springer*; 2nd ed.

[2] J.B., Rawlings (2000) Tutorial overview of Model Predictive Control, *IEEE Control Systems Magazine*, 20 No. 3, pp. 38-52

[3] S.J., Qin and T.A., Badgwell (2003) A survey of industrial model predictive control technology. *Control Engineering Practice* 11(7), 733-764

[4] B., Aufderheide and B.W., Bequette (2001) A variably tuned multiple model predictive controller based on minimal process knowledge. *In proc. of the American Control Conference*, pp. 3490-3495, Arlington

[5] Z., Wan and M.V. Kothare (2004) Efficient scheduled stabilizing output feedback model predictive control for constrained nonlinear systems. *IEEE Transactions on Automatic Control*, 49(7), 1172-1177

[6] A. Rahideh and M.H. Shaheed (2012) Constrained output feedback model predictive control for nonlinear systems *Control Engineering Practice* 20(2012) pp. 431-443

[7] R., De Keyser (2003) Model based Predictive Control for Linear Systems. UNESCO Encyclopaedia of Life Support Systems http://www.eolss.net. Article contribution 6.43.16.1 (available online at: http://www.eolss.net/sample-chapters/c18/e6-43-16-01.pdf). Eolss Publishers Co Ltd, Oxford, 35 pages.

[8] J.M. Maciejowski (2002) Predictive Control with Constraints *Pearson Education Limited*, Edinburgh.

[9] M. Sanchez and J. Rodellar (1996) Adaptive Predictive Control ISBN 0135148618, *Prentice Hall London*

[10] R. De Keyser and C.M. Ionescu (2006) FRTool: a frequency response tool for CACSD in Matlab, *in Proc. of the IEEE Conf. on Computer Aided Control Systems Design*, Munich, Germany, 2006, 2275-2280.