energies

Article

Nonlinear Adaptive Control of Heat Transfer Fluid Temperature in a Parabolic Trough Solar Power Plant

Antonio Nevado Reviriego [1,*,†], Félix Hernández-del-Olmo [2] and Lourdes Álvarez-Barcia [3]

[1] Departamento Ingeniería Eléctrica, Eletrónica y de Control, Universidad Nacional de Educación a Distancia (UNED), 28040 Madrid, Spain
[2] Departamento Inteligencia Artificial, UNED, 28040 Madrid, Spain; felixh@dia.uned.es
[3] Departamento Ingeniería Eléctrica, Eletrónica, Computadoras y Sistemas, Universidad de Oviedo, 33203 Gijón, Spain; lurdesalbar@gmail.com
* Correspondence: anevado@ieec.uned.es; Tel.: +34-91-398-9389
† Current address: Escuela Técnica Superior de Ingenieros Industriales, UNED, Juan del Rosal 12, Ciudad Universitaria, 28040 Madrid, Spain.

Received: 17 July 2017; Accepted: 3 August 2017; Published: 7 August 2017

Abstract: Control of highly nonlinear processes such as solar collector fields is usually a challenging task. A common approach to this problem involves deploying a set of operation point range-specific controllers, whose actions are to be combined in a switching strategy. Discontinuities in control actions upon switching may lead to instabilities and, therefore, achieving bumpless transitions is always a concern. In addition, linear adaptive predictive controllers need to cope with nonlinearities by using high adaptation speeds, often leading to model vulnerability in the presence of aggressive perturbations. Finally, most of the proposed solutions rely on complex plant model developments. In this work, a multivariable nonlinear model-based adaptive predictive controller has been developed and tested against a parabolic trough solar power plant simulation. Since the model employed by this controller accounts for process nonlinearities, adaptation speed can be dramatically reduced, therefore increasing model robustness. The controller is easily initialized and is able to identify and track the process dynamics, including its nonlinearities as it evolves with time, thus requiring neither process up-front modeling nor switching. The presented controller outperforms its linear counterpart both in terms of accuracy and robustness and, due to the generality of its design, it is expected to be applicable to a wide class of linear and nonlinear processes.

Keywords: adaptive control; predictive control; nonlinear control; neural control; parabolic trough

1. Introduction

Distributed solar collector field plants have always posed major control problems when addressing field outlet fluid temperatures. These systems concentrate sunlight over a pipe inside which Heat Transfer Fluid (HTF) flows, accumulating energy and transferring it to another location, where it can be used. The fast, unpredictable, time-varying characteristic of sun irradiation and the inherently nonlinear nature of heat transfer in combination with the transport of the HTF make outlet temperature control a very challenging problem.

It has been almost two decades since this process was characterized and an adaptive predictive control-based solution was proposed in [1]. In that paper, the controller parameters of linear process models were estimated by a recursive least squares algorithm. The authors presented a combination of adaptive predictive techniques with so-called switching control, a methodology in which the controller used at a particular time is selected from a set of running controllers by a supervisor algorithm. In this way, different linear models are used to estimate a nonlinear process, covering a different process operation point range in each model. Switching control has been in the literature for many years while different switching strategies have been proposed and improved [2–4].

A single linear adaptive predictive controller could offer an alternative to switching control although, in that case, the required adaptation speed would need to be high, in order to be able to quickly adapt not only to the time-varying nature of the process, but also to every new operating point when it changes, as seen in [1,5]. High adaptation speeds, however, could make this controller model vulnerable to noise and to unmeasured perturbations.

Apart from adaptive control, most of the documented approaches to this problem [6,7] have required a reasonably accurate model of the system. These solutions, revised in [8,9], include conventional Proportional-Integral-Derivative (PID) control, sometimes presented in conjunction with feed-forward or cascade control, linear and nonlinear model predictive control, optimal control, etc. In practice, however, an appropriate model is seldom available, which, in any case, would be an expensive component.

This paper presents a new Multi-Input Single-Output (MISO) nonlinear adaptive predictive controller for HTF temperature control in a parabolic trough solar plant. The controller is initialized with a simplified linear transfer function of the plant requiring no model development whatsoever, and it is able to adapt to the real plant by itself, keeping track of plant dynamics as it evolves during operation. Since process dynamics are learnt by the controller as a nonlinear function of the operating point, the controller allows for fast and efficient operation point changes even when very low adaptation speeds are used. This fact makes the solution very robust against noise and perturbations. Additionally, since no control action switching is involved, transition discontinuities are not a concern with this controller. The controller has been tested against a solar field simulation recently published in [10] and is presented here as a control proposal for these kinds of plants.

The rest of this paper is organized as follows. Section 2 describes the controller structure, the various modules it comprises and the way they interact with each other. Then, Section 3 provides a description of the experiments conducted with the presented algorithm, along with corresponding discussion and proposed future works. Finally, the conclusions are presented in Section 4.

2. Design

The objective of a predictive control scheme is to make the process output under control follow a desired trajectory. In it, a predictive algorithm calculates the control signal, based on a process model, with the aim of making the predicted process output equal to a desired output trajectory. Since these calculations strongly rely on the process model, predictive control is sometimes combined with an adaptive algorithm aimed at estimating the process model or making it track possible changes in process behaviour.

The structure of the controller presented in this paper derives from the adaptive predictive control scheme implemented for different industries and applications in [11–13]. The controllers described in those papers employ, for the purpose of prediction, a linear process model of the form:

$$y(k) = \sum_{i=1}^{n} a_i(k)y(k-i) + \sum_{i=1}^{m} b_i(k)u(k-i) + \sum_{i=1}^{l} c_i(k)d(k-i), \tag{1}$$

where $y(k)$ is the process output at time k; $u(k)$ represents the process input and $d(k)$, a measurable disturbance, all of which are incremental. The parameters a_i, b_i and c_i are updated at every iteration by an adaptation mechanism with the objective of driving the estimation error towards zero. In Equation (1), the maximum number of parameters for every set is defined by n, m and l. The Equation (1) can be easily extended to consider more measurable disturbances or process inputs and outputs.

This structure, when enabled with the appropriate adaptation algorithm, enables the controller to cope with time-varying processes. Processes showing mild nonlinearities could also be controlled with it, but changes in operating point would always require the controller to re-adapt to the new operation point, making it somehow inefficient, especially when abrupt set point changes are performed.

The controller design presented in this paper considers an adaptation mechanism that includes a model parameter dependency with a set of variables $\boldsymbol{v} = [v_1, v_2, ..., v_q]$ in the form:

$$y(k) = \sum_{i=1}^{n} a_i(k, \boldsymbol{v})y(k-i) + \sum_{i=1}^{m} b_i(k, \boldsymbol{v})u(k-i) + \sum_{i=1}^{l} c_i(k, \boldsymbol{v})d(k-i). \tag{2}$$

At each execution time k and depending on the current measured values of $\boldsymbol{v}$, $\boldsymbol{v}_{meas}$, the adaptation mechanism determines an appropriate set of model parameters $\theta(k, \boldsymbol{v}_{meas}) = [\boldsymbol{a}(k, \boldsymbol{v}_{meas}), \boldsymbol{b}(k, \boldsymbol{v}_{meas}), \boldsymbol{c}(k, \boldsymbol{v}_{meas})]$ to be used for prediction. It is relevant to note that, due to the parameter dependency with $\boldsymbol{v}$, the model in Equation (2) can describe a class of nonlinear processes even in the absence of adaptation. The controller based on this model will be therefore referred to as nonlinear in contrast to the one based on the model shown in Equation (1), which is linear in the absence of adaptation. As it can be seen below, this results in a controller that is able to track the time-varying process dynamics while accounting for its nonlinear characteristics.

2.1. Adaptation

The adaptation mechanism described below comprises an initial static linear model of the form $\theta_0 = [\boldsymbol{a}_0, \boldsymbol{b}_0, \boldsymbol{c}_0]$ plus a $\boldsymbol{v}$-dependent, nonlinear, time-varying function ψ, yielding at time k:

$$\theta(k, \boldsymbol{v}) = \theta_0 + \psi(k, \boldsymbol{v}). \tag{3}$$

This structure allows for a basic model initialization achieved by selecting an appropriate initial linear model θ_0 in conjunction with an initial $\psi(0, \boldsymbol{v}) = 0$. After initialization, the nonlinear estimation $\psi(k, \boldsymbol{v})$ will evolve with time tracking the process dynamics and nonlinearities.

In order for the adaptation mechanism to continuously estimate the required $\psi \in \mathbb{R}^q \mapsto \mathbb{R}^{n+m+l}$ function, a set of $n + m + l$ Gaussian Radial Basis Function (RBF) neural networks are used [14]. An RBF neural network computes its output as a weighted sum of a set of p radial basis functions evaluated at the network input point. The corresponding network weights $w(k)$ are modified as the network learns with the objective of reducing the network error. In the controller presented here, each normalized RBF network has the form:

$$\psi_j(\boldsymbol{v}) = \frac{\sum_p w_{jp}\, e^{-\beta||\boldsymbol{v}-\boldsymbol{c_p}||^2}}{\sum_p e^{-\beta||\boldsymbol{v}-\boldsymbol{c_p}||^2}}, \quad 1 \le j \le n+m+l, \tag{4}$$

where the basis function centers $\boldsymbol{c_p}$ are uniformly spaced over the variable ranges. Figure 1 shows an example of a network for the uni-dimensional case, $q = 1$, comprising fifteen basis functions generated with $\beta = 200$. In the figure, component functions are shown with dashed blue lines, whereas the network output is shown by a solid red line.

In this context, an RBF-based model update requires a pair $(\boldsymbol{v}, \psi(k))$. Whereas the first element is simply measured, the second one must be estimated from a process output measurement $y_{meas}(k)$. This estimation is performed based on the existing error $[y_{meas}(k) - \phi(k-1)^T\theta(k-1)\mid_{\boldsymbol{v}_{meas}}]$ where:

$$\phi(k) = [y(k-1),\ ...,\ y(k-n),\ u(k-1),\ ...,\ u(k-m),\ d(k-1),\ ...,\ d(k-l)]. \tag{5}$$

By using the projection algorithm derived in [15],

$$\theta(k) = \theta(k-1)\mid_{\boldsymbol{v}_{meas}} + \gamma\frac{\phi(k-1)}{\varepsilon + \phi(k-1)^T\phi(k-1)}[y_{meas}(k) - \phi(k-1)^T\theta(k-1)\mid_{\boldsymbol{v}_{meas}}], \tag{6}$$

where ε is a positive small number to avoid division by zero and $0 < \gamma \le 1$ is the adaptation speed.

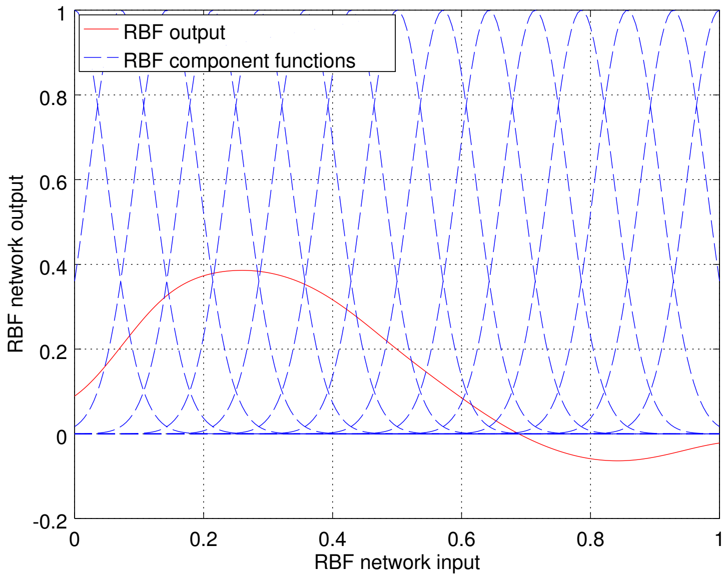

Figure 1. RBF network example providing ψ_j as a function of v_1.

Once the new estimation $\psi(k)$ is available as the difference $\theta(k) - \theta_0$, the RBF networks can be updated. Since the parameter β and the set of centers $\boldsymbol{c_p}$ in Equation (4) are set beforehand and fixed during operation, updating the corresponding set of weights $\boldsymbol{w_j} = [w_{j1}, w_{j2}, ..., w_{jp_{max}}]$ for a given network output $\psi_j(k)$ is a linear regression problem. Therefore, projection can be subsequently applied at full speed to every RBF network separately:

$$\boldsymbol{w_j}(k) = \boldsymbol{w_j}(k-1) + \left\{ \frac{\varphi_j(\boldsymbol{v})}{\varepsilon + \varphi_j(\boldsymbol{v})^T \varphi_j(\boldsymbol{v})} \left[\psi_j(k) - \frac{\sum_p w_{jp}(k-1)\, e^{-\beta||\boldsymbol{v}-\boldsymbol{c_p}||^2}}{\sum_p e^{-\beta||\boldsymbol{v}-\boldsymbol{c_p}||^2}} \right] \right\} \Bigg|_{\boldsymbol{v}=\boldsymbol{v}_{meas}}, \tag{7}$$

where:

$$\varphi_j(\boldsymbol{v}) = \frac{1}{\sum_p e^{-\beta||\boldsymbol{v}-\boldsymbol{c_p}||^2}} \left[e^{-\beta||\boldsymbol{v}-\boldsymbol{c_1}||^2}, e^{-\beta||\boldsymbol{v}-\boldsymbol{c_2}||^2}, ..., e^{-\beta||\boldsymbol{v}-\boldsymbol{c_{p_{max}}}||^2} \right]. \tag{8}$$

It is worth mentioning that, according to Equations (7) and (8), a network update generated by a pair $(\boldsymbol{v}_{meas}, \psi)$ will only modify the estimation of $\psi(\boldsymbol{v})$ in the vicinity of $\boldsymbol{v}_{meas}$, leaving further areas of the hyperspace intact. This property, often referred to as localization, combined with the projection algorithm described above, makes this network an appropriate choice for process dynamics tracking.

In the absence of noise, the projection algorithm shown in Equation (6) has been proven to reduce, on every step, the norm of the model parametric identification error $||\theta_a - \theta(k)||$, θ_a being the actual process parameter set. For noisy environments, a minimum absolute value of the estimation error $[y_{meas}(k) - \phi(k-1)^T \theta(k-1)\,|_{\boldsymbol{v}_{meas}}]$ can be found as a condition for adaptation execution, which guarantees the required parametric identification error reduction. Corresponding proofs can be found in [16]. It must be noted though that, for the sake of simplicity, conditional adaptation has not been implemented in this paper.

Likewise, projection convergence properties directly apply for individual model parameters in the nonlinear RBF adaptation algorithm shown in Equation (7). Indeed, for a given set of variables $\boldsymbol{v}_{meas}$, the projection algorithm is guaranteed to improve the latest model parameter estimation previously found in the vicinity of that point.

2.2. Prediction

The prediction algorithm implemented in this controller is based on Equation (1). Using this equation recursively with the corresponding set of model parameters $\theta(k)$ provided by the adaptation mechanism allows for the prediction in an extended interval $[k,\ k+\lambda]$ of a sequence of future process outputs $\hat{y}(k+h)$. This predicted sequence is obtained as a function of a future control sequence $\hat{u}(k+h-1)$ and a future estimated disturbance sequence $\hat{d}(k+h-1)$ in which $h=[1,\lambda]$. The control action calculation, as described in [11], relies on the following two conditions:

1. The controller output at time k, $\hat{u}(k)$, must be the first value of a predicted controller output sequence in which all the increments in the interval $[k+1, k+\lambda-1]$ are equal to zero. In the same interval, all the increments of the predicted disturbance sequence are assumed to be equal to zero.
2. The predicted process output trajectory must be at time $k+\lambda$ equal to a desired output value $y_d(k+\lambda)$.

The desired trajectory used for the control action calculation is generated from the controller set point r at each time step by a driver block defined by a second order system, in the form:

$$y_d(k+h) = \sum_{i=1}^{2} \alpha_i\, g(k+h-i) + \sum_{i=1}^{2} \beta_i\, r(k+h-i), \tag{9}$$

where:

$$h=[1,\lambda] \quad \text{and} \quad g(\tau) = \begin{cases} y(\tau), & \text{if } \tau \le k, \\ y_d(\tau), & \text{otherwise.} \end{cases} \tag{10}$$

The driver block in Equation (9) thus generates a complete λ instants long desired trajectory $y_d(k)$ in which, as stated in Equation (10), past values of it are replaced by their corresponding actual measurements $y(k)$. The equation coefficients α_i and β_i are chosen so that the resulting relationship has unitary gain.

Finally, the control action generated by this controller is incrementally limited in order to avoid aggresive inputs coming into the plant. Consequently, two absolute consecutive control actions will never differ in more than u_{max}.

2.3. Initialization

Since the RBF-based estimation $\psi(\boldsymbol{v})$ can be easily initialized to $\psi(\boldsymbol{v})=0$ simply by setting every weight vector $\boldsymbol{w}_j$ to $\mathbf{0}$, it is highly desirable to set a realistic model linearization at some mid range point of $\boldsymbol{v}$, θ_0, before the algorithm starts operating. By doing this, the expected performance of the resulting controller will be improved upon start-up.

In order to find a static linear model for initialization θ_0, the algorithm presented in this paper is first started in a degraded mode, as described below:

1. A generic, critically damped, second order linear model $\theta_{0\_}$ is chosen. A high gain of this model is desirable since it will avoid abrupt control actions during the first control instants.
2. The controller is then started with the RBF network disabled and $\theta_{0\_}$ is updated at every control step according to Equation (6).
3. After the linear model $\theta_{0\_}$ has somehow stabilized, it is set as θ_0 and the regular operation described above in this section is started.

2.4. Sequence of Operations

The general sequence of operations, once the regular operation has started, is described below:

1. Obtain a measurement of the process output $y_{meas}(k)$, along with a measurement of the variable vector $\boldsymbol{v}_{meas}$.
2. Calculate the RBF network value at $\boldsymbol{v}_{meas}$, $\psi(\boldsymbol{v}_{meas})$, by using Equation (4).

3. Obtain the expected process model linearization at v_{meas}, $\theta(k-1)|_{v=v_{meas}}$ according to Equation (3).
4. Use Equations (6) and (7) to update the RBF network.
5. Obtain the new expected process model linearization at v_{meas}, $\theta(k)|_{v=v_{meas}}$ according to Equation (3).
6. Calculate the desired output process trajectory y_d by using Equation (9).
7. Calculate the control sequence as described in Section 2.2.
8. Apply the first element of the calculated control sequence.
9. Wait for the selected control period and repeat this sequence from the beginning.

3. Results and Discussion

In order to evaluate the presented algorithm, its performance is compared in this section to the one achieved with the linear version of the controller while controlling the simulation of an HTF system. It must be noted that the linear version of the controller corresponds to the initial degraded mode of the nonlinear controller referred to in Section 2.3.

3.1. Simulation

The simulation of the HTF system used for the evaluation of the controller presented in this paper has been extensively described in [10]. The model describes the HTF system of the parabolic trough collector solar plant "La Africana" in Córdoba, Spain. In the plant, the HTF system comprises 168 loops, containing 48 solar collector elements each. Each collector element, in turn, comprises 28 mirrors together with the corresponding absorber pipe, which is located at the mirror's focal axis. These pipes receive heat from the mirrors and transfer it to the HTF circulating inside them.

The simulation receives from the controller the required HTF mass flow rate and considers mass and energy conservation principles on the HTF system to calculate the heat absorbed by the fluid and its final temperature. Mirror efficiency and thermal losses from the HTF to the surrounding air due to radiation and convection through the pipe walls are considered as well. Therefore, the model considers two inputs, namely the incoming HTF temperature and the HTF mass flow rate; and a single output, the outgoing HTF temperature. Additionally, the simulation considers three disturbances: the Direct Normal Irradiation (DNI), the angle of incidence of the sun and the ambient temperature. In the experiments described below, the only perturbation in use is the DNI, while the other two are kept constant at the maximum angle of incidence and 25 °C, respectively. Additionally, the incoming HTF temperature is also kept constant at a value of 292 °C. As a result, the obtained nonlinear simulation calculates the outgoing HTF temperature based on previous DNI evolutions and HTF mass flow rate set points, which are being generated by the controller under test as its control action.

3.2. Control Scenario

In the paragraphs below, both linear and nonlinear controllers with different adaptation speeds are compared in two common control scenarios. The main objective is to find out whether the nonlinear controller allows for using the adaptation to cope with the potential slow-speed time-varying nature of the process exclusively, since it is not necessary for dealing with the process nonlinearities. This fact would allow the use of smaller γ values in the nonlinear controller during normal control operation, resulting in a much more robust adaptation model.

In the first control scenario defined, the final HTF temperature is driven by the controller under test using a constant set point of 393 °C. All the perturbations and parameters included in the simulation such as the ambient temperature and the angle of incidence of the sun are considered constant, except for the DNI, which evolves following a pre-fixed pattern. This pattern was generated by using a random walk with zero mean and a standard deviation of 3 W/m^2 and it is shown below in the comparison section. The pattern was filtered in order to make it start and stop smoothly, and mirrored afterwards and it is not meant to resemble any real situation; instead, it is aimed at providing a comparison framework where the controllers are sufficiently challenged. Filtering provides DNI stability before and after the tests, while mirroring allows for easy detection of direction-dependent controller responses. During every simulation experiment, discrete DNI values from the pattern are fed into the continuous simulation at a rate of one value per simulated second.

The second control scenario has been targeted to compare the adaptive model robustness in the presence of aggressive perturbations. The experiment simulates a broken wire event while the process is under control by dropping the HTF final temperature measured by the controller to zero during fifteen consecutive control periods. During that time, the controllers are expected to drive the HTF pumps to their minimum value, in an attempt to recover the HTF temperature. The temperature will rise considerably above the set point, but the controllers will keep measuring zero, which will obviously affect their models. After fifteen control periods, the HTF temperature communication will be restored and the controllers will have to drive an unusually high temperature to its set point, with a model that may have been degraded during the first phase of the experiment.

3.3. Controller Configuration and Start-Up

The controller configuration chosen for the experiments presented in this section is described below:

- The controller sampling time has been set to 180 s.
- The variable under control, or process output, referred to as y in Equation (2) corresponds to the HTF final temperature and its set point has been set to 393 °C.
- The control output, or process input, referred to as u in Equation (2) corresponds to the HTF pumps flow rate control loop set point in kg/s.
- The control action incremental limit u_{max} has been set to 300 kg/s.
- The measurable disturbance referred to as d in Equation (2) corresponds to the measured DNI in W/m^2.
- The model size in Equation (2) has been set with the values $m = 2$, $n = 7$ and $l = 7$.
- The horizon length referred to as λ in Equation (2) has been set to 7.
- The model parameters dependency set of variables referred to as v in Equation (2) comprises only the circulating HTF mass flow rate as v_1.

After executing the controller start-up procedure described in Section 2.3, the linear model parameters obtained are the following:

$$\begin{aligned} a_i &= [0.74 \quad -0.35], \\ b_i &= [0.23 \quad 1.69 \quad 0.93 \quad 0.19 \quad 0.27 \quad 0.21 \quad 0.08], \\ c_i &= [0.0 \quad 0.11 \quad 0.78 \quad 0.49 \quad 0.11 \quad 0.12 \quad 0.08]. \end{aligned} \tag{11}$$

This model is therefore saved and used as θ_0 in Equation (3) before the controller starts operating in regular mode. Figure 2 depicts some RBF network outputs after executing the controller in regular mode for some time. Note that the HTF mass flow rate is normalized in the abscissa axis.

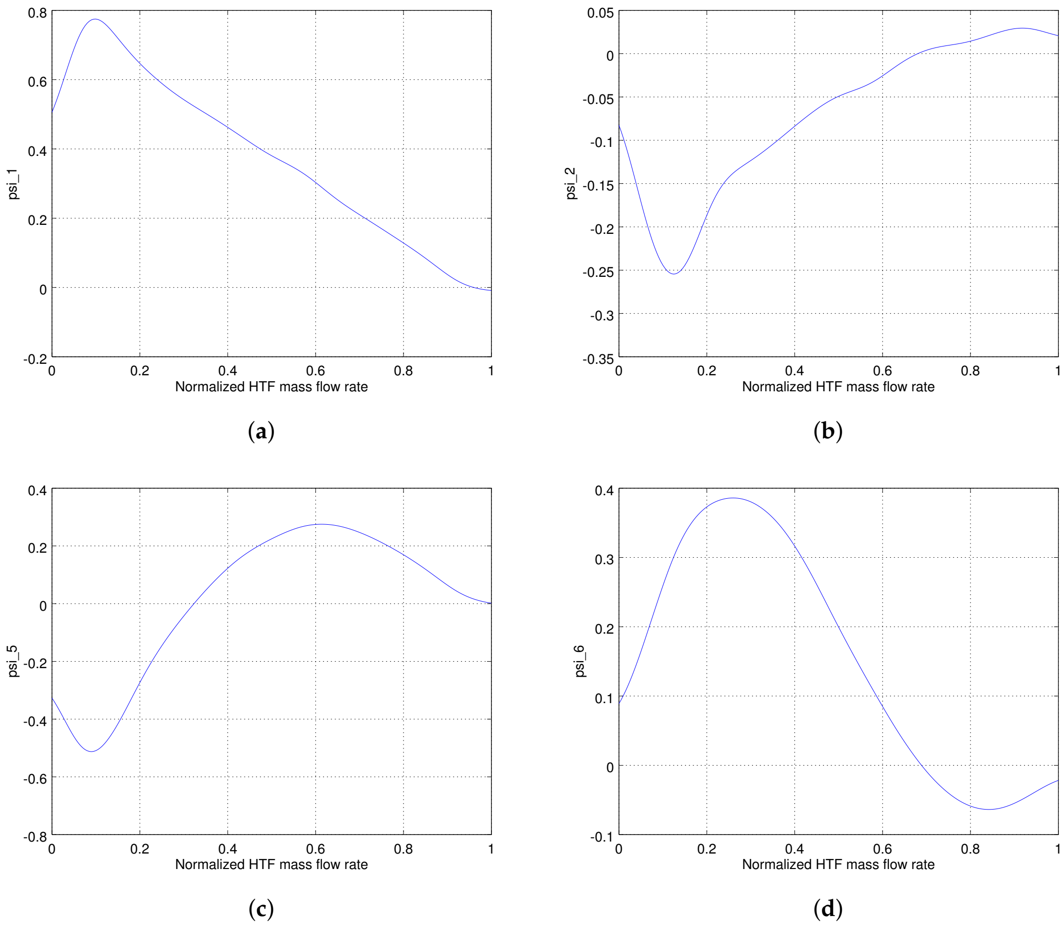

Figure 2. RBF network output examples as a function of normalized HTF mass flow rate. (**a**) ψ_1; (**b**) ψ_2; (**c**) ψ_5; and (**d**) ψ_6.

3.4. Performance Comparison

The paragraphs below present the results of the comparisons carried out with the two control scenarios mentioned above.

3.4.1. Regular Operation

Linear and nonlinear controller operation performance comparisons with different adaptation speeds γ are shown in Figure 3b–f. In those figures, dashed blue lines show the controlled temperatures under the RBF-enabled version of the controller, whereas solid green lines represent the controlled temperatures when the linear model-based operation is used. Temperature set points are the same in both cases and are shown in red. As stated above, Figure 3a shows the DNI pattern employed for comparison, which is common to all the cases presented here.

In the figures, it can be observed how the linear controller can only operate with high adaptation speeds, becoming increasingly unstable as they are reduced. This behaviour accounts for the fact that adaptation is needed to cope with the process nonlinearities. The nonlinear controller, on the other hand, can easily operate with lower speeds, which means that the process nonlinearities are effectively learnt and included into the process model.

In order to summarize this performance comparison, Table 1 shows the HTF final temperature variance for different adaptation speeds and controller modes.

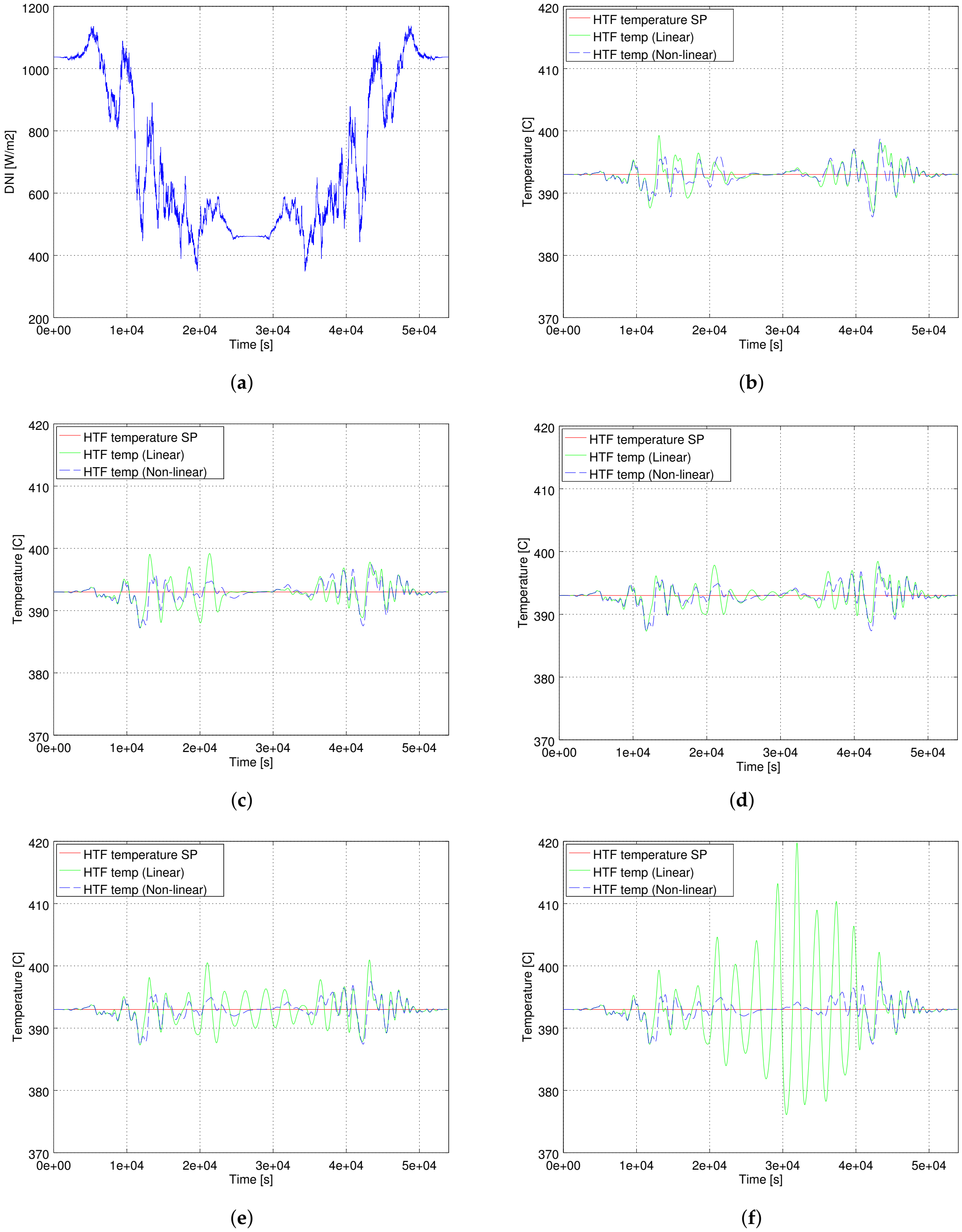

Figure 3. DNI pattern in (**a**) and linear (solid) vs. nonlinear (dashed) controller operation with different adaptation speeds in (**b**–**f**). Time is shown in abscissa. (**a**) DNI pattern; (**b**) $\gamma = 0.7$; (**c**) $\gamma = 0.1$; (**d**) $\gamma = 0.01$; (**e**) $\gamma = 10^{-3}$; and (**f**) $\gamma = 10^{-4}$.

Table 1. Final HTF temperature variance comparison.

Variance	$\gamma = 0.7$	$\gamma = 0.1$	$\gamma = 0.01$	$\gamma = 10^{-3}$	$\gamma = 10^{-4}$
Linear model	2.911	3.506	2.748	4.944	40.751
Nonlinear model	2.407	2.361	2.355	2.338	2.336

For the nonlinear case, the temperature variance is reduced with the adaptation speed. This fact again shows that adaptation is not needed for the controller to cope with process nonlinearities. Even with the highest adaptation speed, the nonlinear controller outperforms the linear controller, since no adaptation is needed to obtain the correct model after every operation point change.

3.4.2. Fault Tolerance

Figure 4 shows the second scenario of the controller comparison described above, in which a fault event has been simulated. HTF temperatures are shown in blue at the upper area of the graphs, whereas control actions are shown in green at the bottom. Thick red lines over real temperatures represent broken wire periods, over which the controllers read null HTF temperatures.

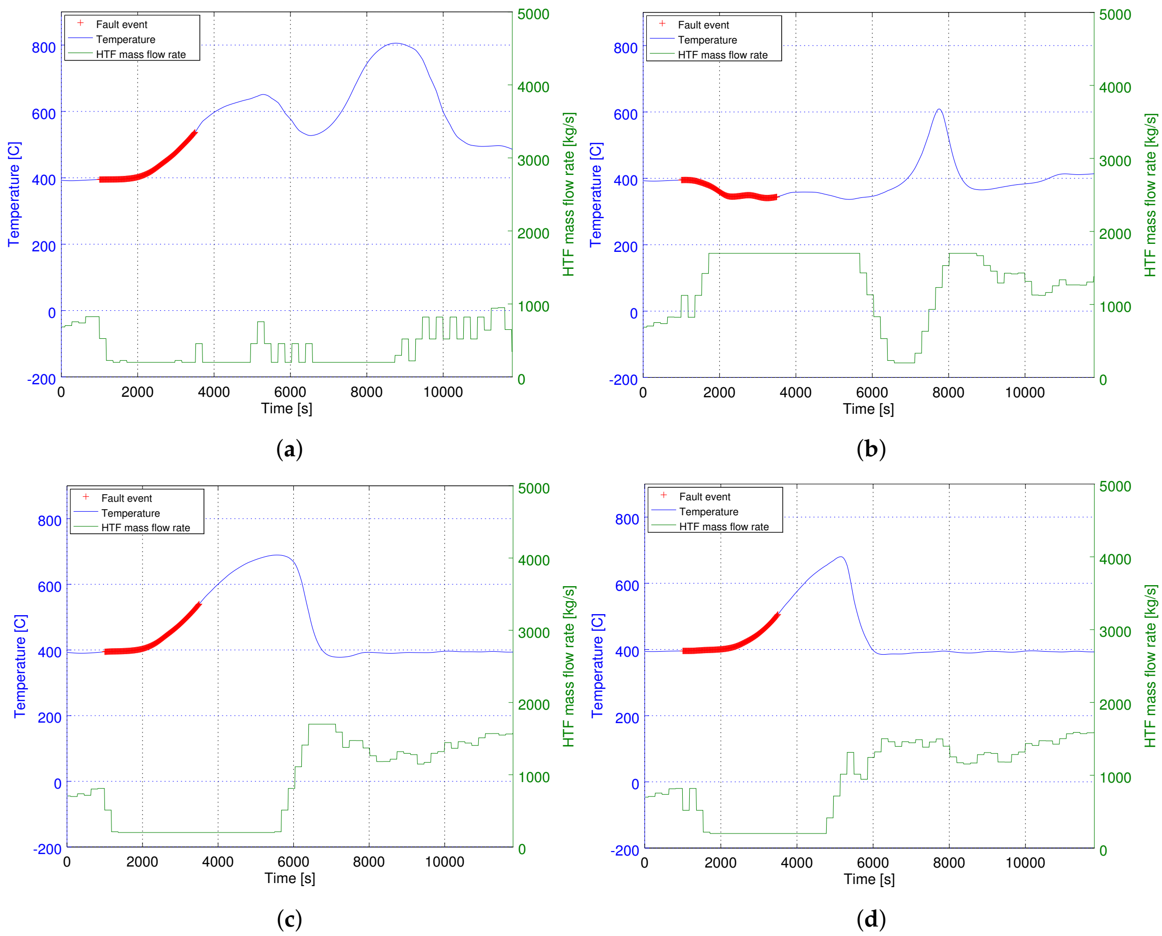

Figure 4. Controller robustness comparison. Linear cases (**a**–**c**) against nonlinear case (**d**). Time is shown in abscissa. (**a**) linear $\gamma = 0.7$; (**b**) linear $\gamma = 0.1$; (**c**) linear $\gamma = 0.01$; and (**d**) nonlinear $\gamma = 10^{-4}$.

It can be appreciated how, for the linear case, the most effective controllers in terms of control performance found in the previous paragraphs and corresponding to high adaptation speeds, are now the most vulnerable. Indeed, the controller with $\gamma = 0.7$ is too damaged for control after this period and do not seem to recover after the fault. In the $\gamma = 0.1$ case, the model is broken in the early stage of the experiment, showing an inverse gain and driving the control action to its maximum value during the fault. The linear $\gamma = 0.01$ case recovers a while after the fault, but, as seen before in Figure 3d, its performance under normal circumstances is not satisfactory, showing oscillations even when DNI is stable around 450 $\mathrm{W/m^2}$.

The nonlinear model, on the other hand, behaves adequately. After the broken wire period is complete, the controller drives the temperature to its set point according to the desired trajectory with no oscillations, recovering the variable control from that point on. Essentially, the nonlinear controller allows for reduced adaptation speeds, resulting in a more robust model while still being able to cope with slow process variations due to aging, wearing, etc.

3.5. Future Works

Future works include the investigation and integration of a method for automatically setting β and $\boldsymbol{c_p}$ in Equation (4). For a given RBF network, these parameters define the maximum degree of curvature that can be reproduced by the network. In order to reduce the controller complexity and to increase its adaptation efficiency, these parameters should be adequately chosen for a particular application. Even if the networks are not too sensitive to these settings, an automated procedure would certainly ease the design phase.

Finally, the generality of the methods employed in this design suggests that it can be directly applied to a wide variety of linear and nonlinear processes. However, this generality must be confirmed in future works with different simulations and processes.

4. Conclusions

In this paper, a multi-input single-output adaptive predictive controller targeted for highly nonlinear processes has been devised. Since the controller can quickly adapt to the process under control, no initial process model is required. Adaptation can also take place online, which makes the controller ideal for time-varying processes.

The model of the controller presented in this paper inherently accounts for process nonlinearities. Therefore, adaptation in this case is intended to cope with the time-varying nature of the process exclusively, and does not require high speeds to control nonlinear processes.

Controller performance and model adaptation robustness have been proven enhanced against those corresponding to the extensively used adaptive predictive controller based on a linear model. Comparisons have been carried out by using an HTF solar heating process simulation, which has been previously presented as an example of a nonlinear process in the literature.

Author Contributions: A.N.R. and F.H.-d.-O. conceived and designed the experiments; A.N.R. and L.Á.-B. performed the experiments; A.N.R. and F.H.-d.-O. analyzed the data; F.H.-d.-O. and L.Á.-B. contributed reagents/materials/analysis tools; A.N.R. wrote the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Rato, L.; Silva, R.; Lemos, J.M.; Coito, F.; Mosca, E.; Borreli, D.; Filatov, N.; Unbehauten, H. Adaptive predictive and switching control of a distributed collector solar field. In Proceedings of the CONTROLO—Portugese Conference on Automatic Control, Coimbra, Portugal, 9–11 September 1998.
2. Leonessa, A.; Haddad, W.M.; Chellaboina, V.S. Nonlinear system stabilization via hierarchical switching control. *IEEE Trans. Autom. Control* **2001**, *46*, 17–28.
3. Pasamontes, M.; Álvarez, J.; Guzmán, J.; Lemos, J.; Berenguel, M. A switching control strategy applied to a solar collector field. *Control Eng. Pract.* **2011**, *19*, 135–145.
4. Zhang, Y.; Chai, T.; Wang, H.; Fu, J.; Zhang, L.; Wang, Y. An adaptive generalized predictive control method for nonlinear systems based on ANFIS and multiple models. *IEEE Trans. Fuzzy Syst.* **2010**, *18*, 1070–1083.

5. Kandare, G.; Nevado, A. Adaptive predictive expert control of dissolved oxygen concentration in a wastewater treatment plant. *Water Sci. Technol.* **2011**, *64*, 1130–1136.
6. Cirre, C.; Valenzuela, L.; Berenguel, M.; Camacho, E. Feedback linearization control for a distributed solar collector field. *Control Eng. Pract.* **2007**, *15*, 1533–1544.
7. Khoukhi, B.; Tadjine, M.; Boucherit, M. Nonlinear continuous-time generalized predictive control of solar power plant. *Int. J. Simul. Multidiscip. Des. Optim.* **2015**, *6*, A3, doi:10.1051/smdo/2015003.
8. Camacho, E.; Rubio, F.; Berenguel, M.; Valenzuela, L. A survey on control schemes for distributed solar collector fields. Part I: Modeling and basic control approaches. *Sol. Energy* **2007**, *81*, 1240–1251.
9. Camacho, E.; Rubio, F.; Berenguel, M.; Valenzuela, L. A survey on control schemes for distributed solar collector fields. Part II: Advanced control approaches. *Sol. Energy* **2007**, *81*, 1252–1272.
10. Barcia, L.A.; Peón, R.; Martínez-Esteban, J.A.; José-Prieto, M.A.; Martín-Ramos, J.A.; Cos Juez, F.J.; Nevado, A. Dynamic modeling of the solar field in parabolic trough solar power plants. *Energies* **2015**, *8*, 13361–13377.
11. Nevado, A.; Martín, I.; Mur, F. Design and application of a steam temperature optimizer in a combined cycle. *Int. J. Adapt. Control Signal Process.* **2012**, *26*, 919–931.
12. Nevado, A.; Requena, R.; Viúdez, D.; Plano, E. A new control strategy for IKN-type coolers. *Control Eng. Pract.* **2011**, *19*, 1066–1074.
13. Kandare, G.; Viúdez-Moreiras, D.; Hernández-del-Olmo, F. Adaptive control of the oxidation ditch reactors in a wastewater treatment plant. *Int. J. Adapt. Control Signal Process.* **2012**, *26*, 976–989.
14. Broomhead, D.S.; Lowe, D. *Radial Basis Functions, Multi-Variable Functional Interpolation and Adaptive Networks*; Royal Signals and Radar Establishment: Malvern, UK, 1988.
15. Goodwin, G.C.; Sin, K.S. *Adaptive Filtering Prediction and Control*; Prentice-Hall: Upper Saddle River, NJ, USA, 1984.
16. Martín-Sánchez, J.M.; Rodellar, J. *Adaptive Predictive Control: From the Concepts to Plant Optimization*; Prentice-Hall International: Upper Saddle River, NJ, USA, 1996.