Chapter 4
MPC of Input-Output Benchmark Wiener Processes

Abstract This Chapter thoroughly discusses implementation details and simulation results of various MPC algorithms introduced in the previous Chapter applied to input-output benchmark processes. Two SISO processes are considered, the second one has complex dynamics and Laguerre parameterisation turns out to be beneficial. Next, three MIMO benchmarks are considered: two with two inputs and two outputs (without and with cross-couplings) and one with as many as ten inputs and two outputs. Implementation details of all algorithms are shortly given. All algorithms are compared in terms of control quality and computational time.

4.1 Simulation Set-Up and Comparison Methodology

All simulations discussed in this book are carried out in MATLAB 2020a. The function `fmincon` is used for nonlinear optimisation, whereas the function `quadprog` is used for quadratic programming. In both cases, the default parameters are used, including stopping criteria.

All MPC algorithms are compared using two performance indices. The first of them

$$E_2 = \sum_{m=1}^{n_y} \sum_{k=1}^{k_{\max}} \left(y_m^{\mathrm{sp}}(k) - y_m(k)\right)^2 \tag{4.1}$$

measures the sum of squared differences between the required set-points, $y_m^{\mathrm{sp}}(k)$, and the actual process outputs, $y_m(k)$, for the whole simulation horizon ($k = 1, \ldots, k_{\max}$)

and for all outputs ($m = 1, \ldots, n_{\mathrm{y}}$). The second one

$$E_{\text{MPC-NO}} = \sum_{m=1}^{n_{\mathrm{y}}} \sum_{k=1}^{k_{\max}} \left(y_m^{\text{MPC-NO}}(k) - y_m(k) \right)^2 \tag{4.2}$$

measures the sum of squared differences between the process outputs controlled by the "ideal" MPC-NO algorithm, $y_n^{\text{MPC-NO}}(k)$, and the process outputs controlled by a compared MPC algorithm, $y_m(k)$. Additionally, the relative calculation time of all MPC algorithms is given. Computation time is measured by means of the functions `tic` and `toc`. As many as 5 repetitions of each simulation scenario are performed, the specified time is calculated as an average of all experiments.

4.2 The SISO Process

4.2.1 Description of the SISO Process

The first considered process is a SISO Wiener system. The linear part of the process (Eqs. (2.1) is of the second order of dynamics ($n_{\mathrm{A}} = n_{\mathrm{B}} = 2$), the coefficients of the model (2.2)-(2.3) are

$$\begin{aligned} a_1 &= -1.4138, & a_2 &= 6.0650 \times 10^{-1} \\ b_1 &= 1.0440 \times 10^{-1}, & b_2 &= 8.8300 \times 10^{-2} \end{aligned} \tag{4.3}$$

The nonlinear static block (Eq. 2.5) is

$$y(k) = g(v(k)) = -\exp(-v(k)) + 1 \tag{4.4}$$

The steady-state characteristic $y(u)$ of the whole Wiener system is depicted in Fig. 4.1.

4.2.2 Implementation of MPC Algorithms for the SISO Process

The following MPC algorithms are compared:

1. The classical LMPC algorithm in which a parameter-constant linear model is used (three example models, obtained for different operating points are considered). The classical Generalized Predictive Control (GPC) algorithm with the DMC disturbance model is used as the LMPC algorithm [15].

2. The classical MPC-inv algorithm (Chapter 3.1). In this approach, the inverse model of the static nonlinear part of the model is used to cancel nonlinearity of the process.
3. Two MPC algorithms with on-line simplified model linearisation and quadratic optimisation: MPC-SSL and MPC-NPSL (Chapter 3.4). The first one uses for free trajectory calculation a linear approximation of the model obtained on-line, in the second one, the full nonlinear Wiener model is used for this purpose.
4. Two MPC algorithms with on-line trajectory linearisation performed once at each sampling instant: MPC-NPLT1 and MPC-NPLT2 (Chapter 3.6). In the first of them, linearisation is carried out along the trajectory of a future sequence of the manipulated variable defined by the manipulated variable applied to the process at the previous sampling instant ($u(k-1)$) as defined by Eq. (3.190). In the second one, the trajectory used for linearisation is defined by the last ($N_{\mathrm{u}}-1$) elements of the optimal input trajectory calculated at the previous sampling instant as defined by Eq. (3.191).
5. The MPC-NPLPT algorithm in which at each sampling instant a few repetitions of trajectory linearisation and optimisation may be necessary, in particular when the process is not close to the required set-point (Chapter 3.8).
6. The best possible MPC-NO algorithm in which the full Wiener model is used for prediction without any simplifications (Chapter 3.2).

All LMPC, MPC-inv, MPC-SSL, MPC-NPSL, MPC-NPLT1, MPC-NPLT2 and MPC-NPLPT algorithms use quadratic optimisation. The MPC-NO algorithm needs on-line nonlinear optimisation at each sampling instant. The LMPC algorithm uses for prediction a linear model; all other MPC algorithms use the same Wiener model, although in different ways.

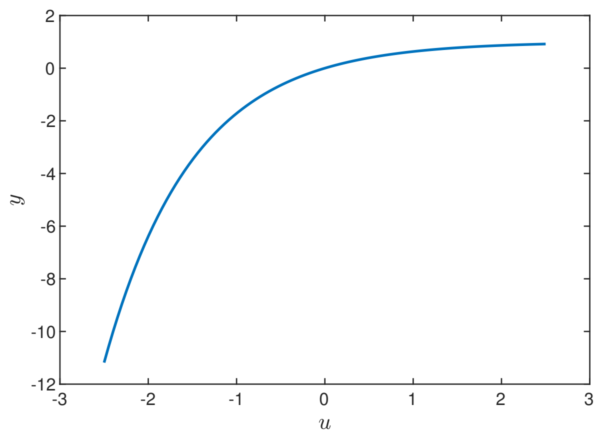

Fig. 4.1 The SISO process: the steady-state characteristic $y(u)$

Next, we shortly detail implementation details of all considered algorithms. In general, all universal equations presented in Chapter 3 are used; here, we only describe specific relations that depend on the static part of the model used.

The parameter-constant linear models used for prediction in the LMPC scheme are obtained for three different operating points: the model 1 for $y = -0.5$, the model 2 for $y = 0$ and the model 3 for $y = 1$. It means that the model actually used in LMPC is the linear dynamic part of the Wiener process multiplied by the gain of the nonlinear static block for the considered operating points. In general, from Eqs. (3.70) and (4.4), we have

$$K = \frac{\mathrm{d}g(v)}{\mathrm{d}v} = \exp(-v) \tag{4.5}$$

We obtain $v = -4.0546 \times 10^{-1}$, $v = 0$ and $v = 6.9315 \times 10^{-1}$ for the operating points 1, 2 and 3, respectively. Hence, the following gains of the nonlinear static blocks are calculated off-line: $K = 1.5$, $K = 1$ and $K = 0.5$.

In a similar way the gain of the nonlinear static block is calculated in the MPC-SSL and MPC-NPSL algorithms, but calculations are performed successively on-line, at each sampling instant. The time-varying gain is

$$K(k) = \frac{\mathrm{d}g(v(k))}{\mathrm{d}v(k)} = \exp(-v(k)) \tag{4.6}$$

where $v(k)$ is the model signal.

In the MPC-NPLT1 and MPC-NPLT2 algorithms, the entries of the derivative matrix $\boldsymbol{H}(k)$ are computed from Eq. (3.211). For the nonlinear block (4.4), we have

$$\frac{\mathrm{d}g(v^{\mathrm{traj}}(k+p|k))}{\mathrm{d}v^{\mathrm{traj}}(k+p|k)} = \exp(-v^{\mathrm{traj}}(k+p|k)) \tag{4.7}$$

Similarly, for calculation of the matrix $\boldsymbol{H}^t(k)$ in the MPC-NPLPT scheme, we use Eq. (3.283). We obtain

$$\frac{\mathrm{d}g(v^{t-1}(k+p|k))}{\mathrm{d}v^{t-1}(k+p|k)} = \exp(-v^{t-1}(k+p|k)) \tag{4.8}$$

A neural network of the MLP type [3, 10, 11, 12, 14] with one hidden layer containing five nonlinear units and a linear output layer is used as the inverse model of the nonlinear static block in the MPC-inv algorithm. The nonlinear units use the tanh activation function. MLP neural networks are used throughout this book because they have the following essential advantages: excellent approximation ability [4], a simple structure and in practice, a low number of parameters (weights) is sufficient to obtain good models.

4.2.3 MPC of the SISO Process

Parameters of all compared MPC algorithms are the same: $N = 10, N_u = 3, \lambda = 0.25$, the constraints imposed on the manipulated variable are: $u^{\min} = -2.5$, $u^{\max} = 2.5$. The horizons are long enough, the coefficient λ is sufficient for nonlinear MPC algorithms. In this book, we do not consider tuning of MPC algorithms, i.e. selection of appropriate horizons and tuning coefficients. These issues are thoroughly discussed in classical textbooks [7, 15], a review of possible approaches is presented in [2]. A practical example of finding the parameters of MPC for a solid oxide fuel cell is described in [6], a similar study concerned with a boiler-turbine unit is reported in [5]. A simple but efficient procedure how to select coefficients of the MPC cost-function is described in [8, 9]. An optimisation-based approach to tuning MPC is discussed in [13]; effectiveness of four global optimisation methods (the Particle Swarm Optimisation (PSO) method, the firefly algorithm, the grey wolf optimiser and the Jaya algorithm) is compared.

In the first part of simulations, the model is perfect (no modelling errors) and the process is not affected by any disturbances. Let us verify performance of the simplest approach to MPC, i.e. the LMPC algorithm in which a parameter-constant linear model is used for prediction (three example models are used, obtained for different operating points). Fig. 4.2 depicts simulation results for a few set-point changes. Unfortunately, the process is nonlinear and the LMPC algorithm does not lead to good control quality. In particular, some oscillations appear after the third set-point change.

Fig. 4.3 compares performance of two simple MPC algorithms with on-line model linearisation, i.e. the MPC-SSL and MPC-NPSL strategies, v.s. the best possible MPC-NO scheme in which the nonlinear Wiener model is used without any simplifications. Both algorithms with model linearisation work much better than the LMPC scheme; there are no oscillations when the set-point changes in a broad range. Application of the nonlinear model for calculation of the free trajectory in the MPC-NPSL algorithm allows to reduce overshoot for negative set-point changes and increase speed for positive ones comparing with the trajectories obtained in the MPC-SSL scheme. Fig. 4.4 depicts changes of the time-varying gain of the nonlinear static block calculated in the MPC-NPSL and MPC-SSL algorithms. Changes in the MPC-NPSL scheme are slower when compared with those observed in the MPC-SSL one.

Fig. 4.5 compares performance of three MPC algorithms with on-line trajectory linearisation, i.e. MPC-NPLT1, MPC-NPLT2 and MPC-NPLPT strategies, v.s. the reference MPC-NO scheme. The algorithms with one linearisation at each sampling instant, i.e. the MPC-NPLT1 and MPC-NPLT2 ones, are better than the MPC-NPSL and MPC-SSL schemes with model linearisation, but still, they give slightly different trajectories than those possible in the MPC-NO algorithm. The MPC-NPLPT algorithm gives practically the same trajectory as the MPC-NO one. The

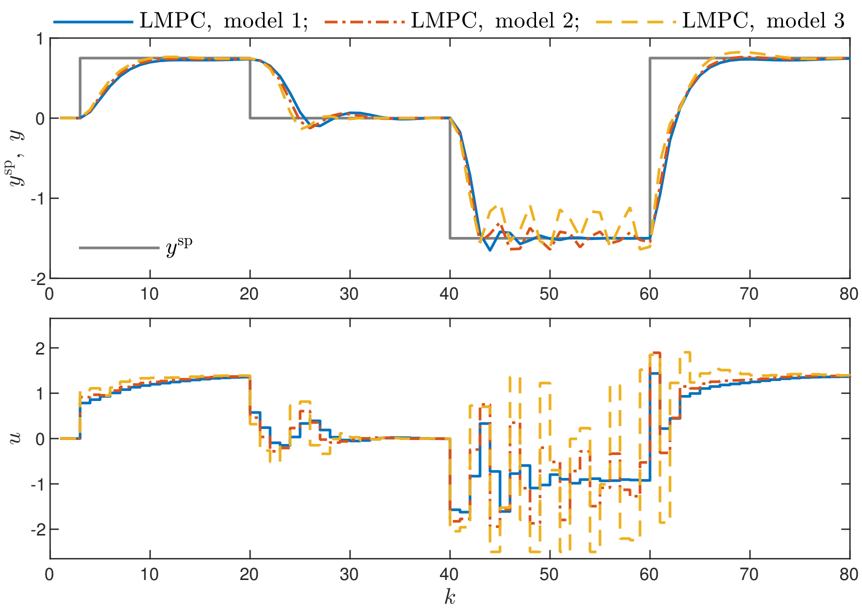

Fig. 4.2 The SISO process: simulation results of the linear LMPC algorithm based on different models, obtained for different operating points

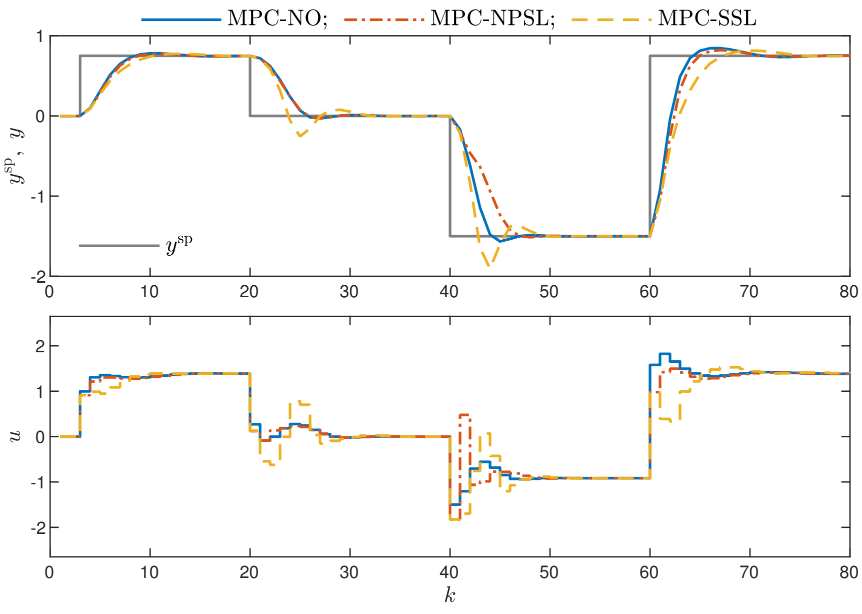

Fig. 4.3 The SISO process: simulation results of the MPC-NO, MPC-NPSL and MPC-SSL algorithms

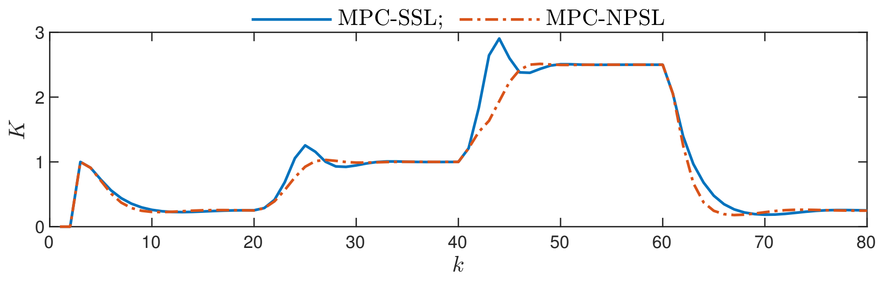

Fig. 4.4 The SISO process: the time-varying gain of the nonlinear static block calculated in the MPC-NPSL and MPC-SSL algorithms

additional parameters of the MPC-NPLPT algorithm are: $\delta = \delta_{\mathrm{u}} = \delta_{\mathrm{y}} = 0.1$, $N_0 = 2$, the maximal number of internal iterations is 5. Fig. 4.6 shows simulation results of the MPC-NPLPT algorithm for two different values of the parameter $\delta = \delta_{\mathrm{u}} = \delta_{\mathrm{y}}$: 1 and 10. It is clear that the greater these parameters, the worse the resulting control quality and the bigger the differences from the trajectories obtained in the best possible MPC-NO strategy. Fig. 4.7 presents the number of internal iterations of the MPC-NPLPT algorithm for different values of the parameter $\delta = \delta_{\mathrm{u}} = \delta_{\mathrm{y}}$. When the process output is close to the required set-point (i.e. the process is close to the steady-state), one internal iteration is sufficient. When a step change of the set-point occurs, the process is far from the steady-state and more than one internal iteration is necessary. The actual number of internal iterations depends on the parameter $\delta = \delta_{\mathrm{u}} = \delta_{\mathrm{y}}$. Of course, the lower that parameter, the higher the number of internal iterations are necessary and they last longer after each set-point step.

Finally, we consider the classical MPC-inv approach to control dynamical systems described by Wiener models based on the inverse model of the static nonlinear part of the model. Simulation results are depicted in Fig. 4.8, the results obtained for the MPC-NPLPT algorithm are given for comparison. For the perfect model and no disturbances, the MPC-inv scheme gives very good results; for three set-points changes, they are even slightly faster than in the case of the MPC-NPLPT algorithm. In one case, the changes are slower. In two cases, overshoot is lower.

All considered MPC algorithms are compared in Table 4.1 in terms of the performance criteria E_2 (Eq. (4.1)) and $E_{\mathrm{MPC\text{-}NO}}$ (Eq. (4.2)). The number of internal iterations necessary in the MPC-NPLPT scheme is specified (they are summarised for the whole simulation horizon). Additionally, the scaled calculation time is given, the result for the most computationally demanding solution, i.e. the MPC-NO strategy, corresponds to 100%. In general, the simple MPC-SSL and MPC-NPSL algorithms with on-line model linearisation give quite good results, but much better accuracy is possible when the algorithms with on-line trajectory linearisation are used. In particular, the MPC-NPLPT algorithm is able to give practically the same control

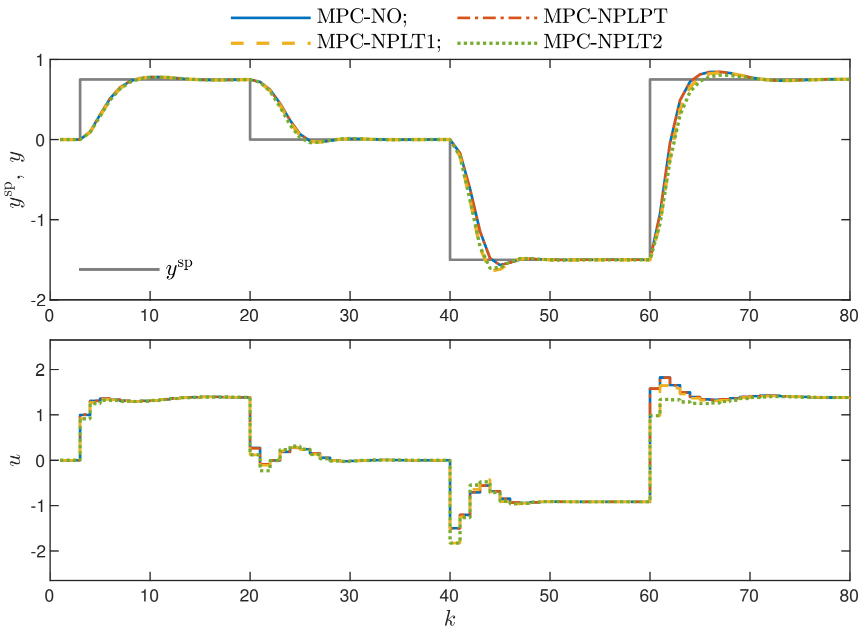

Fig. 4.5 The SISO process: simulation results of the MPC-NO, MPC-NPLPT, MPC-NPLT1 and MPC-NPLT2 algorithms

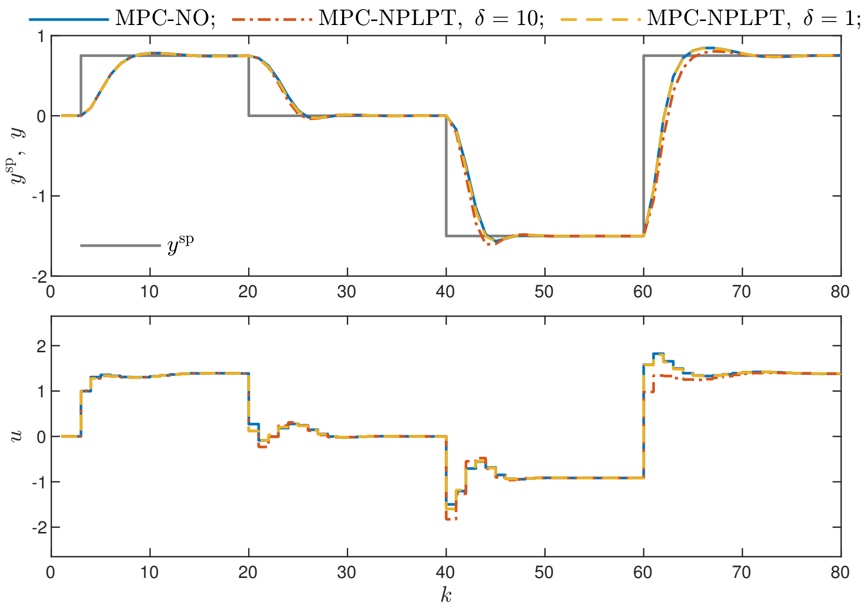

Fig. 4.6 The SISO process: simulation results of the MPC-NO and MPC-NPLPT algorithms for different values of the parameter $\delta = \delta_u = \delta_y$

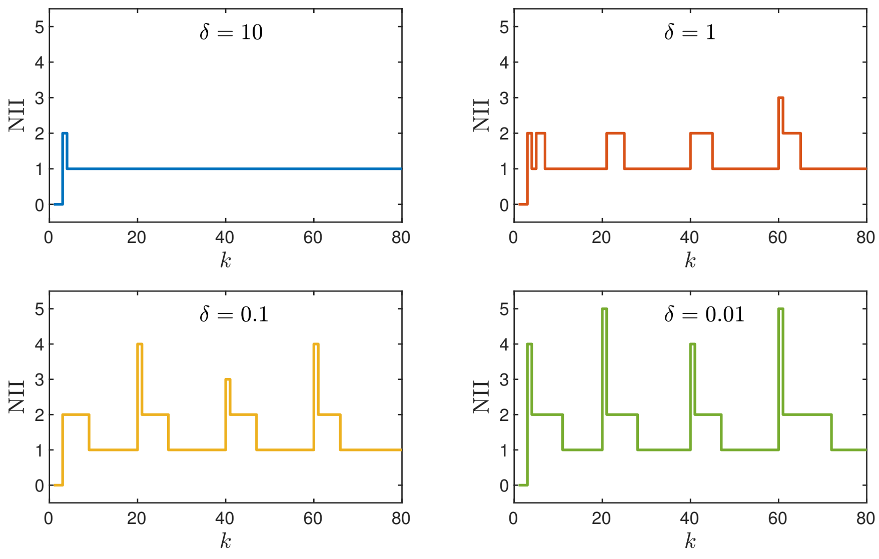

Fig. 4.7 The SISO process: the number of internal iterations (NII) of the MPC-NPLPT algorithm for different values of the parameter $\delta = \delta_{\mathrm{u}} = \delta_{\mathrm{y}}$

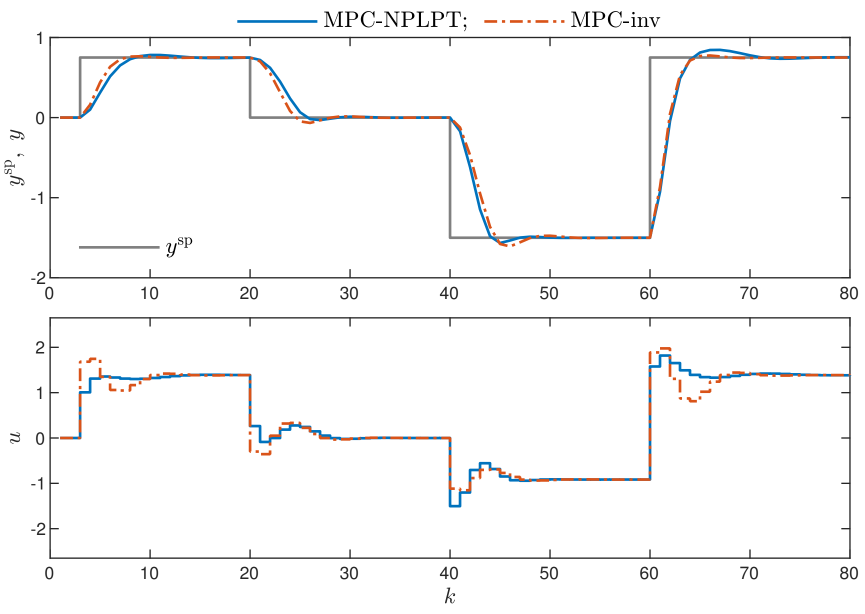

Fig. 4.8 The SISO process: simulation results of the MPC-NPLPT and MPC-inv algorithms

Table 4.1 The SISO process: comparison of all considered MPC algorithms in terms of the control performance criteria (E_2 and $E_{\text{MPC-NO}}$), the sum of internal iterations (SII) and the calculation time

Algorithm	E_2	$E_{\text{MPC-NO}}$	SII	Calculation time (%)
LMPC, model 1	1.7682×10^1	7.5352×10^{-1}	–	36.7
LMPC, model 2	1.6884×10^1	8.5626×10^{-1}	–	36.7
LMPC, model 3	1.6612×10^1	2.0040	–	36.8
MPC-inv	1.6272×10^1	2.3381×10^{-1}	–	39.3
MPC-SSL	1.7964×10^1	1.4605	–	38.8
MPC-NPSL	1.8641×10^1	8.0533×10^{-1}	–	38.7
MPC-NPLT1	1.6917×10^1	1.6185×10^{-1}	–	42.7
MPC-NPLT2	1.7006×10^1	2.1282×10^{-1}	–	40.9
MPC-NPLPT, $\delta = 10$	1.6968×10^1	2.1156×10^{-1}	79	42.1
MPC-NPLPT, $\delta = 1$	1.6363×10^1	4.7067×10^{-3}	96	46.1
MPC-NPLPT, $\delta = 10^{-1}$	1.6513×10^1	2.6108×10^{-5}	109	48.7
MPC-NPLPT, $\delta = 10^{-2}$	1.6524×10^1	5.3482×10^{-7}	123	51.6
MPC-NPLPT, $\delta = 10^{-3}$	1.6523×10^1	3.4081×10^{-7}	132	54.4
MPC-NPLPT, $\delta = 10^{-4}$	1.6523×10^1	3.4171×10^{-7}	149	57.8
MPC-NPLPT, $\delta = 10^{-5}$	1.6523×10^1	4.4881×10^{-7}	151	58.8
MPC-NO	1.6524×10^1	–	–	100.0

accuracy as the reference MPC-NO strategy because the obtained index $E_{\text{MPC-NO}}$ is very close to 0. Of course, the lower the parameters $\delta = \delta_{\text{u}} = \delta_{\text{y}}$, the higher the number of the internal iterations and the better the control accuracy. It is interesting to consider the computational time of the compared algorithms. The simple MPC-SSL and MPC-NPSL algorithms with on-line model linearisation need practically the same calculation time as the LMPC strategy (lower than 40% of that necessary in the MPC-NO scheme), the MPC algorithms with trajectory linearisation are more demanding. Of course, the lower the parameters $\delta = \delta_{\text{u}} = \delta_{\text{y}}$, the longer the calculation time. It is important to stress the fact that the MPC-NPLPT scheme for $\delta = \delta_{\text{u}} = \delta_{\text{y}} = 0.1$, which gives practically the same trajectories as the MPC-NO one (Fig. 4.5), is characterised by the calculation time lower than 50% of that necessary in the MPC-NO scheme.

It is interesting to study how the computational time is influenced by the length of prediction and control horizons for all considered MPC algorithms. Table 4.2 presents the comparison in such a way that the calculation time for the horizons $N = 10$, $N_{\text{u}} = 3$, corresponds to 100%. The first important observation is that the control horizon has a major impact on the calculation time, the prediction one has much a lower influence. It is natural since the control horizon defines the number of decision variables in MPC optimisation. The second observation is that all MPC algorithms with on-line linearisation are significantly less computationally demanding for different combinations of horizons than the MPC-NO one. The best results are obtained for long control horizons, e.g. $N_{\text{u}} = 10$. In such cases, the MPC-NO scheme requires 2-3 longer calculation time.