Chapter 3
MPC Algorithms Using Input-Output Wiener Models

Please cite the book:
Maciej Ławryńczuk: Nonlinear Predictive Control Using Wiener Models: Computationally Efficient Approaches for Polynomial and Neural Structures. Studies in Systems, Decision and Control, vol. 389, Springer, Cham, 2022.

Abstract This Chapter details MPC algorithms for processes described by input-output Wiener models. At first, the simple MPC-inv method based on the inverse static model is recalled. The rudimentary MPC algorithm with Nonlinear Optimisation (MPC-NO) repeated at each sampling instant is described. Next, two computationally efficient MPC methods with on-line model linearisation are characterised: the MPC scheme with Simplified Successive Linearisation (MPC-SSL) and the MPC approach with Nonlinear Prediction and Simplified Linearisation (MPC-NPSL). Two MPC schemes with on-line trajectory linearisation are also detailed: the MPC method with Nonlinear Prediction and Linearisation along the Trajectory (MPC-NPLT) and the MPC scheme with Nonlinear Prediction and Linearisation along the Predicted Trajectory (MPC-NPLPT). All discussed MPC algorithms are first presented in their rudimentary versions; the variants with parameterisation using Laguerre functions, which reduces the number of decision variables, are described next.

3.1 MPC-inv Algorithm

The simplest and the most frequent approach to control dynamical processes described by Wiener models is to use an inverse static model of the nonlinear part of the Wiener model and a linear control algorithm [4]. The role of the inverse model is to try to cancel the nonlinear static behaviour of the controlled process. The resulting control system structure is depicted in Fig. 3.1. In the SISO case, the inverse model is

$$v(k) = \tilde{g}(y(k)) \tag{3.1}$$

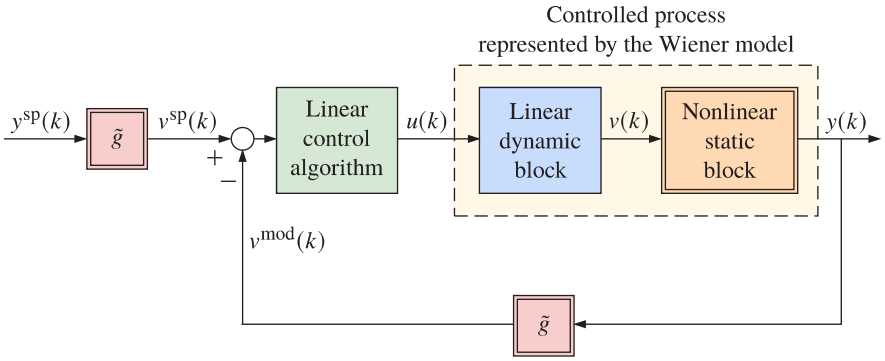

Fig. 3.1 The control system structure for dynamical processes described by Wiener models using the inverse static model; $\tilde{g}$ denotes the inverse static model

where the general function $\tilde{g}\colon \mathbb{R} \to \mathbb{R}$. The inverse model is used twice for control. Firstly, it is used to calculate the value of the auxiliary model signal basing on the measured process output

$$v^{\mathrm{mod}}(k) = \tilde{g}(y(k)) \tag{3.2}$$

Typically, the real signal v does not exist in the controlled process and its measurement is impossible. Secondly, the inverse model is necessary to calculate the value of the auxiliary variable corresponding to the current set-point of the controlled variable

$$v^{\mathrm{sp}}(k) = \tilde{g}(y^{\mathrm{sp}}(k)) \tag{3.3}$$

From the perspective of the control algorithm based on a linear model, the controlled output variable is v, not y.

The described approach may be used with different kinds of linear controllers, not only MPC ones [1, 2, 5, 16, 17, 18] but also with PID or Internal Model Control (IMC) [3, 6] ones. An alternative is to use dynamic inversion [19]. Although the whole idea seems to be simple and potentially efficient, it has important drawbacks:

1. The inverse model must exist, which means that the controller based on the inverse model cannot be used when it is impossible to find the inverse representation of the nonlinear part of the Wiener model. Of course, the problem occurs when the static characteristic of the nonlinear block is non-invertible. Furthermore, as it is discussed in Chapter 4.6, in some cases, the inverse models may be very complex.
2. As it will be shown in Chapter 4, the described structure may result in control accuracy worse than in some other approaches especially developed for dynamical processes described by the Wiener model. In particular, it may give unacceptable control quality when the model used in MPC is not perfect and/or the process is affected by unmeasured disturbances. Typically, both static and dynamic properties of processes are nonlinear. Hence, nonlinear process behaviour cannot be entirely cancelled by a nonlinear static block.

Depending on the dimensionality of the process and model structure, we may distinguish the following cases:

1. In the simplest case, when the controlled process is described by the SISO Wiener model shown in Fig. 2.1, it is only necessary to find a SISO inverse model (3.1).
2. For MIMO Wiener models I and III, depicted in Figs. 2.2 and 2.4, respectively, we have to use as many as n_{y} inverse SISO models

$$v_1(k) = \tilde{g}_1(y_1(k)) \tag{3.4}$$

$$\vdots$$

$$v_{n_{\mathrm{y}}}(k) = \tilde{g}_{n_{\mathrm{y}}}(y_{n_{\mathrm{y}}}(k)) \tag{3.5}$$

3. For MIMO Wiener models II and IV, shown in Figs. 2.3 and 2.5, respectively, we have to use n_{v} inverse MISO models

$$v_1(k) = \tilde{g}_1(y_1(k), \ldots, y_{n_{\mathrm{y}}}(k)) \tag{3.6}$$

$$\vdots$$

$$v_{n_{\mathrm{v}}}(k) = \tilde{g}_{n_{\mathrm{v}}}(y_1(k), \ldots, y_{n_{\mathrm{y}}}(k)) \tag{3.7}$$

   In this case, the inverse models are likely to be complicated, in particular when there are really many process outputs.
4. For the MIMO Wiener model V shown in Fig. 2.6, it is necessary to use as many as $n_{\mathrm{u}}n_{\mathrm{y}}$ inverse models

$$v_{m,n}(k) = \tilde{g}_{m,n}(y_m(k)) \tag{3.8}$$

   for all $m = 1, \ldots, n_{\mathrm{y}}, n = 1, \ldots, n_{\mathrm{u}}$. In this case, we require that the inverse models calculate all $n_{\mathrm{u}}n_{\mathrm{y}}$ auxiliary signals on the basis of only n_{y} process output signals. It may turn out to be very difficult or even impossible.

All things considered, provided that the inverse model exists, the MPC-inv algorithm may be used in the SISO case or in the MIMO case when all nonlinear static blocks are of SISO type, i.e. when the MIMO Wiener models I or III are used. When the MIMO Wiener models II, IV or V are used, the inverse models may be very complicated, which makes implementation difficult or impossible.

3.2 MPC-NO Algorithm

In the MPC algorithm with Nonlinear Optimisation (MPC-NO), the decision variables, i.e. the future increments of the manipulated variable(s) (1.3) are calculated at each sampling instant k from the optimisation problem. In the SISO case, the formulation (1.12) is used, whereas in the general MIMO one, the optimisation task is (1.20). In all cases, it may be transformed to a compact vector-matrix form (1.35). When the soft constraints are imposed on the controlled variables, the opti-

misation problem is defined by Eq. (1.38), which may be transformed to a compact vector-matrix form (1.39).

The model of the controlled process is used to calculate the predicted values of the controlled variables for the consecutive sampling instants over the prediction horizon, i.e. the quantities $\hat{y}(k+1|k), \ldots, \hat{y}(k+N|k)$. Provided that we have a perfect model, in the SISO case, the prediction equation for the sampling instant $k+p$ is

$$\hat{y}(k+p|k) = y(k+p|k) \tag{3.9}$$

where the symbol $y(k+p|k)$ denotes the output of the model for the sampling instant $k+p$ used at the current instant k. Unfortunately, for prediction calculation we must take into account that usually the model used in MPC is not perfect, i.e. there are differences between properties of the process and its model and that the measurement of the process output is not ideal. In order to compensate for all these factors, the following general prediction equation must be used [15, 20]

$$\hat{y}(k+p|k) = y(k+p|k) + d(k) \tag{3.10}$$

where $d(k)$ is the current estimation of the unmeasured disturbance which acts on the process output. In the most typical approach (named "the DMC disturbance model"), it is assumed that the disturbance is constant over the whole prediction horizon and its value is determined as the difference between the real (measured) value of the process output ($y(k)$) and the model output ($y^{\mathrm{mod}}(k)$)

$$d(k) = y(k) - y^{\mathrm{mod}}(k) \tag{3.11}$$

It may be easily proved that when the unmeasured disturbance estimation is used in the prediction equation, the MPC algorithm has the integral action which leads to no steady-state error [15, 20].

In the MIMO case, the predictions are

$$\hat{y}_m(k+p|k) = y_m(k+p|k) + d_m(k) \tag{3.12}$$

for all process outputs, i.e. for $m = 1, \ldots, n_{\mathrm{y}}$. Disturbance estimations are

$$d_m(k) = y_m(k) - y_m^{\mathrm{mod}}(k) \tag{3.13}$$

Prediction Using SISO Wiener Model

At first, let us discuss the SISO case in which the Wiener model depicted in Fig. 2.1 is used. Using the general prediction equation (3.10) and from the description of the nonlinear static block, i.e. Eq. (2.5), we have

$$\hat{y}(k+p|k) = g(v(k+p|k)) + d(k) \tag{3.14}$$

where $p = 1, \ldots, N$. From the description of the linear dynamic block, i.e. Eq. (2.4), we have

$$\begin{aligned} v(k+1|k) = {} & b_1 u(k|k) + b_2 u(k-1) + b_3 u(k-2) + \ldots \\ & + b_{n_\mathrm{B}} u(k - n_\mathrm{B} + 1) \\ & - a_1 v(k) - a_2 v(k-1) - a_3 v(k-2) - \ldots \\ & - a_{n_\mathrm{A}} v(k - n_\mathrm{A} + 1) \end{aligned} \tag{3.15}$$

$$\begin{aligned} v(k+2|k) = {} & b_1 u(k+1|k) + b_2 u(k|k) + b_3 u(k-1) + \ldots \\ & + b_{n_\mathrm{B}} u(k - n_\mathrm{B} + 2) \\ & - a_1 v(k+1|k) - a_2 v(k) - a_3 v(k-1) - \ldots \\ & - a_{n_\mathrm{A}} v(k - n_\mathrm{A} + 2) \end{aligned} \tag{3.16}$$

$$\begin{aligned} v(k+3|k) = {} & b_1 u(k+2|k) + b_2 u(k+1|k) + b_3 u(k|k) + \ldots \\ & + b_{n_\mathrm{B}} u(k - n_\mathrm{B} + 3) \\ & - a_1 v(k+2|k) - a_2 v(k+1|k) - a_3 v(k) - \ldots \\ & - a_{n_\mathrm{A}} v(k - n_\mathrm{A} + 3) \end{aligned} \tag{3.17}$$

$$\vdots$$

In general, Eqs. (3.15)-(3.17) may be rewritten in the following compact form

$$\begin{aligned} v(k+p|k) = {} & \sum_{i=1}^{I_{\mathrm{uf}}(p)} b_i u(k-i+p|k) + \sum_{i=I_{\mathrm{uf}}(p)+1}^{n_\mathrm{B}} b_i u(k-i+p) \\ & - \sum_{i=1}^{I_{\mathrm{vf}}(p)} a_i v(k-i+p|k) - \sum_{i=I_{\mathrm{vf}}(p)+1}^{n_\mathrm{A}} a_i v(k-i+p) \end{aligned} \tag{3.18}$$

for $p = 1, \ldots, N$. Taking into account the prediction of the auxiliary variable for the future sampling instant $k + p$ performed at the current instant k, the number of future manipulated variables, from the sampling instant k, i.e. $u(k|k), u(k+1|k), \ldots$, is denoted by

$$I_{\mathrm{uf}}(p) = \max(\min(p, n_\mathrm{B}), 0) \tag{3.19}$$

The number of future values of the signal v, from the sampling instant $k + 1$, i.e. $v(k+1|k), v(k+2|k), \ldots$, is

$$I_{\mathrm{vf}}(p) = \min(p - 1, n_\mathrm{A}) \tag{3.20}$$

The quantity $v(k)$ does not depend on the future manipulated variables but only on past ones, i.e. up to the sampling instant $k - 1$, it is clear from Eq. (2.4). From Eqs. (2.6) and (3.11), the unmeasured disturbance is estimated from

$$d(k) = y(k) - g\left(\sum_{i=1}^{n_\mathrm{B}} b_i u(k-i) - \sum_{i=1}^{n_\mathrm{A}} a_i v(k-i)\right) \tag{3.21}$$

Prediction Using MIMO Wiener Model I

Next, we will discuss the MIMO case in which the first structure of the Wiener model depicted in Fig. 2.2 is used. Using the general prediction equation (3.12) and from Eq. (2.14), we have

$$\hat{y}_m(k+p|k) = g_m(v_m(k+p|k)) + d_m(k) \tag{3.22}$$

where $m = 1, \ldots, n_y$, $p = 1, \ldots, N$. Using the vector notation, Eq. (3.14) may be obtained, the same that is used in the SISO case (in such a case, all three components are vectors of length n_y). Next, from Eq. (2.11), we have

$$\begin{aligned} v_m(k+p|k) = \sum_{n=1}^{n_u} \left(\sum_{i=1}^{I_{uf}(p)} b_i^{m,n} u_n(k-i+p|k) + \sum_{i=I_{uf}(p)+1}^{n_B} b_i^{m,n} u_n(k-i+p) \right) \\ - \sum_{i=1}^{I_{vf}(p)} a_i^m v_m(k-i+p|k) - \sum_{i=I_{vf}(p)+1}^{n_A} a_i^m v_m(k-i+p) \end{aligned} \tag{3.23}$$

where $m = 1, \ldots, n_y$, $p = 1, \ldots, N$. The quantities $I_{uf}(p)$ and $I_{vf}(p)$ (Eqs. (3.19) and (3.20)) are independent of the model input and output because all model channels have the same order of dynamics, defined by the same values of n_A and n_B. From Eqs. (2.17) and (3.13), the unmeasured disturbances are estimated from

$$d_m(k) = y_m(k) - g_m \left(\sum_{n=1}^{n_u} \sum_{i=1}^{n_B} b_i^{m,n} u_n(k-i) - \sum_{i=1}^{n_A} a_i^m v_m(k-i) \right) \tag{3.24}$$

Prediction Using MIMO Wiener Model II

Next, we will discuss the MIMO case in which the second structure of the Wiener model depicted in Fig. 2.3 is used. Using the general prediction equation (3.12) and from Eq. (2.25), we have

$$\hat{y}_m(k+p|k) = g_m(v_1(k+p|k), \ldots, v_{n_v}(k+p|k)) + d_m(k) \tag{3.25}$$

where $m = 1, \ldots, n_y$, $p = 1, \ldots, N$. The signals $v_m(k+p|k)$ are calculated from Eq. (3.23), in a similar way it is done for the MIMO Wiener model I, for $p = 1, \ldots, N$, but now $m = 1, \ldots, n_v$. From Eqs. (2.28) and (3.13), the unmeasured disturbances

are estimated from

$$d_m(k) = y_m(k) - g_m\left(\sum_{n=1}^{n_u}\sum_{i=1}^{n_B} b_i^{1,n}u_n(k-i) - \sum_{i=1}^{n_A} a_i^1 v_1(k-i), \ldots, \sum_{n=1}^{n_u}\sum_{i=1}^{n_B} b_i^{n_v,n}u_n(k-i) - \sum_{i=1}^{n_A} a_i^{n_v} v_{n_v}(k-i)\right) \tag{3.26}$$

Prediction Using MIMO Wiener Model III

In the case of the third structure of the MIMO Wiener model depicted in Fig. 2.4, using the general prediction equation (3.12) and from Eq. (2.53), we have

$$\hat{y}_m(k+p|k) = g_m\left(\sum_{n=1}^{n_u} v_{m,n}(k+p|k)\right) + d_m(k) \tag{3.27}$$

for $m = 1, \ldots, n_y$, $p = 1, \ldots, N$. From Eq. (2.47), we have

$$v_{m,n}(k+p|k) = \sum_{i=1}^{I_{uf}(m,n,p)} b_i^{m,n}u_n(k-i+p|k) + \sum_{i=I_{uf}(m,n,p)+1}^{n_B^{m,n}} b_i^{m,n}u_n(k-i+p) - \sum_{i=1}^{I_{vf}(m,n,p)} a_i^{m,n}v_{m,n}(k-i+p|k) - \sum_{i=I_{vf}(m,n,p)+1}^{n_A^{m,n}} a_i^{m,n}v_{m,n}(k-i+p) \tag{3.28}$$

for all $m = 1, \ldots, n_y$, $n = 1, \ldots, n_u$, $p = 1, \ldots, N$. Because transfer functions of the consecutive input-output channels may have different order of dynamics, in place of Eqs. (3.19)-(3.20), we use

$$I_{uf}(m,n,p) = \max(\min(p, n_B^{m,n}), 0) \tag{3.29}$$

and

$$I_{vf}(m,n,p) = \min(p-1, n_A^{m,n}) \tag{3.30}$$

From Eqs. (2.56) and (3.13), the unmeasured disturbances are estimated from

$$d_m(k) = y_m(k) - g_m\left(\sum_{n=1}^{n_u}\left(\sum_{i=1}^{n_B^{m,n}} b_i^{m,n}u_n(k-i) - \sum_{i=1}^{n_A^{m,n}} a_i^{m,n}v_{m,n}(k-i)\right)\right) \tag{3.31}$$

Prediction Using MIMO Wiener Model IV

In the case of the fourth structure of the MIMO Wiener model depicted in Fig. 2.5, using the general prediction equation (3.12) and from Eq. (2.68), we have

$$\hat{y}_m(k+p|k) = g_m\left(\sum_{n=1}^{n_\mathrm{u}} v_{1,n}(k+p|k), \ldots, \sum_{n=1}^{n_\mathrm{u}} v_{n_\mathrm{v},n}(k+p|k)\right) + d_m(k) \tag{3.32}$$

for $m = 1, \ldots, n_\mathrm{y}$, $p = 1, \ldots, N$. The signals $v_{m,n}(k+p|k)$ are calculated from Eq. (3.28), in a similar way it is done in the case of the MIMO model III, for $n = 1, \ldots, n_\mathrm{u}$, $p = 1, \ldots, N$, but now $m = 1, \ldots, n_\mathrm{v}$. From Eqs. (2.71) and (3.13), the unmeasured disturbances are estimated from

$$\begin{aligned} d_m(k) = y_m(k) - g_m\Bigg(&\sum_{n=1}^{n_\mathrm{u}}\left(\sum_{i=1}^{n_\mathrm{B}^{1,n}} b_i^{1,n} u_n(k-i) - \sum_{i=1}^{n_\mathrm{A}^{1,n}} a_i^{1,n} v_{1,n}(k-i)\right), \ldots, \\ &\sum_{n=1}^{n_\mathrm{u}}\left(\sum_{i=1}^{n_\mathrm{B}^{n_\mathrm{v},n}} b_i^{n_\mathrm{v},n} u_n(k-i) - \sum_{i=1}^{n_\mathrm{A}^{n_\mathrm{v},n}} a_i^{n_\mathrm{v},n} v_{n_\mathrm{v},n}(k-i)\right)\Bigg) \end{aligned} \tag{3.33}$$

Prediction Using MIMO Wiener Model V

In the case of the fifth structure of the MIMO Wiener model depicted in Fig. 2.6, using the general prediction equation (3.12) and from Eq. (2.77), we have

$$\hat{y}_m(k+p|k) = \sum_{n=1}^{n_\mathrm{u}} g_{m,n}(v_{m,n}(k+p|k)) + d_m(k) \tag{3.34}$$

for $m = 1, \ldots, n_\mathrm{y}$, $p = 1, \ldots, N$. The signals $v_{m,n}(k+p|k)$ are calculated from Eq. (3.28), in the same way it is done in the case of the MIMO models III, for all $m = 1, \ldots, n_\mathrm{y}$, $n = 1, \ldots, n_\mathrm{u}$, $p = 1, \ldots, N$. From Eqs. (2.80) and (3.13), the unmeasured disturbances are estimated from

$$d_m(k) = y_m(k) - \sum_{n=1}^{n_\mathrm{u}} g_{m,n}\left(\sum_{i=1}^{n_\mathrm{B}^{m,n}} b_i^{m,n} u_n(k-i) - \sum_{i=1}^{n_\mathrm{A}^{m,n}} a_i^{m,n} v_{m,n}(k-i)\right) \tag{3.35}$$

Optimisation

One may easily note from Eqs. (3.14), (3.22), (3.25), (3.27), (3.32) and (3.34) that the predicted values of the controlled variables are nonlinear functions of the future values of the manipulated variable(s), or in different words, nonlinear functions of the

calculated future increments (1.3). Hence, the optimisation problems (1.12), (1.20), (1.35), (1.38) and (1.39) are in fact nonlinear ones. Hence, a nonlinear optimisation method must be used on-line in the MPC-NO approach.

At first let us discuss how the MPC-NO optimisation problem with hard output constraints defined by Eq. (1.35) should be reformulated in order to solve it in MATLAB. We will use the `fmincon` function for optimisation. Its syntax is

```
X = fmincon(FUN,X0,A,B,Aeq,Beq,LB,UB,NONLCON,OPTIONS)
```

It solves the general nonlinear optimisation task

$$\begin{aligned} &\min_{\boldsymbol{x}(k)} \{f(\boldsymbol{x}(k))\} \\ &\text{subject to} \\ &\boldsymbol{A}\boldsymbol{x}(k) \leq \boldsymbol{B}(k) \\ &\boldsymbol{A}_{\text{eq}}\boldsymbol{x}(k) = \boldsymbol{B}_{\text{eq}} \\ &\boldsymbol{C}(\boldsymbol{x}(k)) \leq \boldsymbol{0} \\ &\boldsymbol{C}_{\text{eq}}(\boldsymbol{x}(k)) = \boldsymbol{0} \\ &\boldsymbol{LB} \leq \boldsymbol{x}(k) \leq \boldsymbol{UB} \end{aligned} \tag{3.36}$$

The `fmincon` function makes it possible to take into account 5 types of constraints: linear inequalities ($\boldsymbol{A}\boldsymbol{x}(k) \leq \boldsymbol{B}(k)$), linear equalities ($\boldsymbol{A}_{\text{eq}}\boldsymbol{x}(k) = \boldsymbol{B}_{\text{eq}}$), nonlinear inequalities ($\boldsymbol{C}(\boldsymbol{x}(k)) \leq \boldsymbol{0}$), nonlinear equalities ($\boldsymbol{C}_{\text{eq}}(\boldsymbol{x}(k)) = \boldsymbol{0}$) and bounds ($\boldsymbol{LB} \leq \boldsymbol{x}(k) \leq \boldsymbol{UB}$). When compared with the general nonlinear optimisation problem (3.36) solved by the `fmincon` function, in our MPC-NO optimisation task (1.35), the decision variable vector is $\boldsymbol{x}(k) = \triangle\boldsymbol{u}(k)$ and there are only three types of constraints: linear inequalities defined by

$$\boldsymbol{A} = \begin{bmatrix} -\boldsymbol{J} \\ \boldsymbol{J} \end{bmatrix}, \ \boldsymbol{B}(k) = \begin{bmatrix} -\boldsymbol{u}^{\min} + \boldsymbol{u}(k-1) \\ \boldsymbol{u}^{\max} - \boldsymbol{u}(k-1) \end{bmatrix} \tag{3.37}$$

nonlinear inequalities defined by

$$\boldsymbol{C}(\boldsymbol{x}(k)) = \begin{bmatrix} -\hat{\boldsymbol{y}}(k) + \boldsymbol{y}^{\min} \\ \hat{\boldsymbol{y}}(k) - \boldsymbol{y}^{\max} \end{bmatrix} \tag{3.38}$$

and bounds defined by

$$\boldsymbol{LB} = \triangle\boldsymbol{u}^{\min}, \ \boldsymbol{UB} = \triangle\boldsymbol{u}^{\max} \tag{3.39}$$

Linear and nonlinear equality constraints are not present in our MPC-NO optimisation problem (1.35). The vector of predicted values of the controlled variable(s), $\hat{\boldsymbol{y}}(k)$, is calculated at each sampling instant for the given vector $\triangle\boldsymbol{u}(k)$ recurrently from Eqs. (3.14) and (3.18) (the SISO Wiener model) or Eqs. (3.22) and (3.23) (the MIMO Wiener model I) or Eqs. (3.23) and (3.25) (the MIMO Wiener model II) or Eqs. (3.27) and (3.28) (the MIMO Wiener model III) or Eqs. (3.28) and (3.32) (the MIMO Wiener model IV) or Eqs. (3.28) and (3.34) (the MIMO Wiener model V).

Let us note that the vector $\hat{\boldsymbol{y}}(k)$ is present in the minimised cost-function and in the output constraints. It means that both of them are nonlinear.

Next, let us consider the MPC-NO optimisation problem with soft output constraints defined by Eq. (1.39). The algorithm calculates at each sampling instant not only the future increments of the manipulated variable(s) but also optimal violations of the original hard output constraints necessary to guarantee feasibility. Hence, the vector of the decision variables

$$\boldsymbol{x}(k) = \begin{bmatrix} \triangle \boldsymbol{u}(k) \\ \varepsilon^{\min}(k) \\ \varepsilon^{\max}(k) \end{bmatrix} \tag{3.40}$$

is of length $n_{\mathrm{u}}N_{\mathrm{u}} + 2n_{\mathrm{y}}$. Let us also define auxiliary matrices

$$\boldsymbol{N}_1 = \begin{bmatrix} \boldsymbol{I}_{n_{\mathrm{u}}N_{\mathrm{u}} \times n_{\mathrm{u}}N_{\mathrm{u}}} & \boldsymbol{0}_{n_{\mathrm{u}}N_{\mathrm{u}} \times 2n_{\mathrm{y}}} \end{bmatrix} \tag{3.41}$$

$$\boldsymbol{N}_2 = \begin{bmatrix} \boldsymbol{0}_{n_{\mathrm{y}} \times n_{\mathrm{u}}N_{\mathrm{u}}} & \boldsymbol{I}_{n_{\mathrm{y}} \times n_{\mathrm{y}}} & \boldsymbol{0}_{n_{\mathrm{y}} \times n_{\mathrm{y}}} \end{bmatrix} \tag{3.42}$$

$$\boldsymbol{N}_3 = \begin{bmatrix} \boldsymbol{0}_{n_{\mathrm{y}} \times (n_{\mathrm{u}}N_{\mathrm{u}} + n_{\mathrm{y}})} & \boldsymbol{I}_{n_{\mathrm{y}} \times n_{\mathrm{y}}} \end{bmatrix} \tag{3.43}$$

which are of dimensionality $n_{\mathrm{u}}N_{\mathrm{u}} \times (n_{\mathrm{u}}N_{\mathrm{u}} + 2n_{\mathrm{y}})$, $n_{\mathrm{y}} \times (n_{\mathrm{u}}N_{\mathrm{u}} + 2n_{\mathrm{y}})$ and $n_{\mathrm{y}} \times (n_{\mathrm{u}}N_{\mathrm{u}} + 2n_{\mathrm{y}})$, respectively. The following relations are true

$$\triangle \boldsymbol{u}(k) = \boldsymbol{N}_1 \boldsymbol{x}(k) \tag{3.44}$$

$$\varepsilon^{\min}(k) = \boldsymbol{N}_2 \boldsymbol{x}(k) \tag{3.45}$$

$$\varepsilon^{\max}(k) = \boldsymbol{N}_3 \boldsymbol{x}(k) \tag{3.46}$$

We will also rewrite the vectors $\underline{\boldsymbol{\varepsilon}}^{\min}(k)$ and $\underline{\boldsymbol{\varepsilon}}^{\max}(k)$ defined by Eqs. (1.40). They may be expressed in the following way

$$\underline{\boldsymbol{\varepsilon}}^{\min}(k) = \boldsymbol{I}_{N \times 1} \otimes \varepsilon^{\min}(k) \tag{3.47}$$

$$\underline{\boldsymbol{\varepsilon}}^{\max}(k) = \boldsymbol{I}_{N \times 1} \otimes \varepsilon^{\max}(k) \tag{3.48}$$

where the symbol $\otimes$ denotes the Kronecker product of two vectors. Using Eqs. (3.45)-(3.46), the relations (3.47)-(3.48) may be expressed as

$$\underline{\boldsymbol{\varepsilon}}^{\min}(k) = \boldsymbol{I}_{N \times 1} \otimes \boldsymbol{N}_2 \boldsymbol{x}(k) \tag{3.49}$$

$$\underline{\boldsymbol{\varepsilon}}^{\max}(k) = \boldsymbol{I}_{N \times 1} \otimes \boldsymbol{N}_3 \boldsymbol{x}(k) \tag{3.50}$$