TWMCC ⋆ Texas-Wisconsin Modeling and Control Consortium

Technical report number 2001-01

The Velocity Algorithm LQR: a survey

Gabriele Pannocchia and James B. Rawlings
Department of Chemical Engineering
University of Wisconsin-Madison
Madison, WI 53706

May 18, 2001

Abstract

In this report the velocity algorithm – the one the computes the input change given the state change and the current measurement – is studied under the general framework of LQ regulators. Similarities and differences with the standard algorithm are discussed in order to emphasize advantages and disadvantages of each formulation.

Keywords

Predictive Control, Linear Quadratic Regulation, Velocity Algorithm, State augmentation

1 Introduction

The velocity algorithm computes, at each time k, the input change Δu_k instead of the input value u_k, using the state change Δx_k instead of the state value x_k. The velocity form is used in the digital implementation of PID controllers (e.g. [5]) in order to avoid wind-up. Moreover, the original industrial formulations of Model Predictive Control [7] [1] [2] computed the input change Δu from a discrete impulse response model. More recently, we can find other examples of the velocity algorithm in [4] [6] [3].

We present this algorithm under the general framework of Linear-Quadratic regulators. We show what conditions should be satisfied in order to obtain a complete equivalence between the velocity algorithm ($\Delta x - \Delta u$) and the traditional algorithm ($x - u$). Moreover, we address cases for which the velocity algorithm cannot be used.

2 Algorithm formulation

2.1 The model

Given a linear time-invariant discrete system defined in terms of state vector $x_k \in \mathbb{R}^n$, input vector $u_k \in \mathbb{R}^m$, output vector $y_k \in \mathbb{R}^p$ and matrices (A, B, C), we consider the following new variables:

$$\Delta x_k = x_k - x_{k-1} \qquad z_k = y_k - y_t \qquad \Delta u_k = u_k - u_{k-1} \tag{1}$$

where y_t is the output target, assumed constant. Next we consider the following augmented state-space model:

$$\tilde{x}_k = \begin{bmatrix} \Delta x_k \\ z_k \end{bmatrix} \quad \tilde{u}_k = \Delta u_k \quad \tilde{A} = \begin{bmatrix} A & 0 \\ CA & I \end{bmatrix} \quad \tilde{B} = \begin{bmatrix} B \\ CB \end{bmatrix} \quad \tilde{C} = \begin{bmatrix} 0 & I \end{bmatrix} \tag{2}$$

which leads to the following new model equation:

$$\begin{aligned} \tilde{x}_{k+1} &= \tilde{A}\tilde{x}_k + \tilde{B}\tilde{u}_k \qquad \tilde{x}_0 \text{ given} \\ z_k &= \tilde{C}\tilde{x}_k \end{aligned} \tag{3}$$

2.2 The regulator

The regulator is formulated as an infinite horizon optimal control problem

$$\min_{\tilde{u}_k} \frac{1}{2} \sum_{k=0}^{\infty} \tilde{x}_k^T \tilde{Q} \tilde{x}_k + \tilde{u}_k^T \tilde{R} \tilde{u}_k \tag{4}$$

subject to eqn. 3, where $\tilde{R}$ is a symmetric positive definite matrix of dimension m, and

$$\tilde{Q} \leftarrow \begin{bmatrix} 0 & 0 \\ 0 & Q \end{bmatrix} \tag{5}$$

in which Q symmetric positive definite matrix of dimension p.

The optimal solution of eqn. 4 is given by

$$\begin{aligned} \tilde{u}_k &= -(\tilde{R} + \tilde{B}^T S \tilde{B})^{-1} \tilde{B}^T S \tilde{A} \tilde{x}_k \\ &= -K_{LQ} \tilde{x}_k \end{aligned} \tag{6}$$

where S is the steady-state solution of the Riccati equation

$$0 = \tilde{A}^T S \tilde{A} - S + \tilde{Q} - \tilde{A}^T S \tilde{B} (\tilde{R} + \tilde{B}^T S \tilde{B})^{-1} \tilde{B}^T S \tilde{A} \tag{7}$$

Once $\tilde{u}_k$ is computed, the effective input injected into the plant is

$$u_k = u_{k-1} + \tilde{u}_k \tag{8}$$

Therefore, the velocity algorithm requires the storage of the previous control action.

2.3 The estimator

The augmented state vector $\tilde{x}_k$, in general, cannot be completely measured, so reconstruction of the state is required. The optimal Kalman filter is chosen and its algorithm is summarized here.

The model equation used by the estimator is

$$\tilde{x}_{k+1} = \tilde{A}\tilde{x}_k + \tilde{B}\tilde{u}_k + \tilde{G}\tilde{w}_k \qquad \tilde{x}_0 \text{ given} \tag{9}$$

where $\tilde{w}_k$ is a zero-mean random sequence with covariance $Q'_{\tilde{w}}$. The initial value is modeled as a Gaussian variable

$$E\{\tilde{x}_0\} = \hat{\tilde{x}}_0 \qquad E\left\{(\tilde{x}_0 - \hat{\tilde{x}}_0)(\tilde{x}_0 - \hat{\tilde{x}}_0)^T\right\} = P_{0|-1} \tag{10}$$

The vector of measurements is

$$y_k - y_t = \tilde{C}\tilde{x}_k + \tilde{v}_k \tag{11}$$

where $\tilde{v}_k$ is zero-mean random sequence with covariance $R_{\tilde{v}}$.

At each time k the measurement y_k is assumed to be known; the prediction of the current state vector $\hat{\tilde{x}}_{k|k-1}$ and its covariance $P_{k|k-1}$, given the model and the previous measurements (see eqs. 14) are known. The Kalman filter gain matrix is given by

$$L_k = P_{k|k-1}\tilde{C}^T\left[\tilde{C}P_{k|k-1}\tilde{C}^T + R_{\tilde{v}}\right]^{-1} \tag{12}$$

Thus, the filtering equations are

$$\begin{aligned} \hat{\tilde{x}}_{k|k} &= \hat{\tilde{x}}_{k|k-1} + L_k\left[(y_k - y_t) - \tilde{C}\hat{\tilde{x}}_{k|k-1}\right] \\ P_{k|k} &= P_{k|k-1} - L_k\tilde{C}P_{k|k-1} \end{aligned} \tag{13}$$

Finally, the future state vector and its covariance matrix are predicted by

$$\begin{aligned} \hat{\tilde{x}}_{k+1|k} &= \tilde{A}\hat{\tilde{x}}_{k|k} + \tilde{B}\tilde{u}_k \\ P_{k+1|k} &= \tilde{A}P_{k|k}\tilde{A}^T + \tilde{G}Q'_{\tilde{w}}\tilde{G}^T \end{aligned} \tag{14}$$

This algorithm is repeated at each time and the updated state vector $\hat{\tilde{x}}_{k|k}$ is used by the regulator in eqn. 6.

3 Key-issues of the velocity algorithm

There are two issues that require a detailed analysis:

- initialization of the model ($\hat{\tilde{x}}_0$ and P_0);
- choice of the noise model ($\tilde{G}$, $Q'_{\tilde{w}}$ and $R_{\tilde{v}}$).

In order to address this problem we derive the velocity algorithm from the traditional formulation (x – u); then we present the equations that permit us to build an algorithm in the traditional form given an algorithm in the velocity form.

In the traditional formulation, the following general model is considered (in which the integrated input and output disturbance cases are covered):

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k + B_d d_k + Gw_k^x \\ d_{k+1} &= d_k + G_d w_k^d \\ y_k &= Cx_k + C_d d_k + v_k \end{aligned} \tag{15}$$

where d is the integrated input - output disturbance, B_d and C_d are matrices of appropriate dimension, w_k^x, w_k^d and v_k are random sequences with the following covariance matrices:

$$E\{w_k^x (w_k^x)^T\} = Q_x \qquad E\{w_k^d (w_k^d)^T\} = Q_d \tag{16a}$$

$$E\{w_k^x (w_k^d)^T\} = Q_{xd} \qquad E\{v_k v_k^T\} = R_v \tag{16b}$$

3.1 Initialization of the model

First, we consider the evolution of standard deterministic systems at time k, given the initial value x_0 and d_0 and remembering that the disturbance is constant, we can write

$$\begin{aligned} x_k &= Ax_{k-1} + B_d d_{k-1} \\ &= A^k x_0 + \left(A^{k-1} + \cdots + I\right) B_d d_0 + \left(A^{k-1} B u_0 + \cdots B u_{k-1}\right) \end{aligned} \tag{17}$$

from which

$$x_k - x_0 = \left(A^k - I\right) x_0 + \left(A^{k-1} + \cdots + I\right) B_d d_0 + A^{k-1} B u_0 + \cdots + B u_{k-1} \tag{18}$$

Then let us consider the evolution of the autonomous systems in the velocity form

$$\begin{aligned} \Delta x_k &= A\Delta x_{k-1} + B\Delta u_{k-1} \\ &= A^k \Delta x_0 + A^{k-1} B \Delta u_0 + \cdots + B\Delta u_{k-1} \end{aligned} \tag{19}$$

from which

$$x_k - x_0 = A\Sigma\Delta x_0 + \Sigma B(u_{-1}) + A^{k-1} B\Delta u_0 + \cdots + B\Delta u_{k-1} \tag{20}$$

in which $\Sigma = \sum_{j=0}^{k-1} A^j$, and u_{-1} is the input at time $k = -1$. Comparing the right hand sides of eqs. 18 and 20, with some matrix algebra we obtain (under the assumption that $(A - I)^{-1}$ exists)

$$A\Delta x_0 = (A - I)\, x_0 + B_d d_0 - B u_{-1} \tag{21}$$

If A^{-1} exists, we can rewrite eqn. 21 as:

$$\Delta x_0 = \left(I - A^{-1}\right) x_0 + A^{-1} B_d d_0 - A^{-1} B u_{-1} \tag{22}$$

Eqn. 21 or 22 is fundamental for the initialization of the algorithm because it ensures that the standard and the velocity models evolve in a congruent manner, *i.e.* they compute the same state x_k at each time. The other equation that is useful for the initialization of the algorithm is

$$z_0 = Cx_0 + C_d d_0 - y_t \tag{23}$$

Given the initial estimates in the traditional form, $\hat{x}_0$ and $\hat{d}_0$ and using eqs 22 and 23, we can compute the initial estimate and its covariance matrix in the velocity form, as reported below:

$$\begin{bmatrix} \Delta\hat{x}_0 \\ \hat{z}_0 \end{bmatrix} = \begin{bmatrix} I - A^{-1} & A^{-1}B_d \\ C & C_d \end{bmatrix} \begin{bmatrix} \hat{x}_0 \\ \hat{d}_0 \end{bmatrix} + \begin{bmatrix} -A^{-1}Bu_{-1} \\ -y_t \end{bmatrix} \tag{24}$$

$$P_{0|-1} = E\left\{ \begin{bmatrix} \Delta x_0 - \Delta\hat{x}_0 \\ z_0 - \hat{z}_0 \end{bmatrix} \begin{bmatrix} \Delta x_0 - \Delta\hat{x}_0 \\ z_0 - \hat{z}_0 \end{bmatrix}^T \right\} = \begin{bmatrix} P_0^{11} & P_0^{12} \\ (P_0^{12})^T & P_0^{22} \end{bmatrix} \tag{25}$$

where

$$\begin{aligned} P_0^{11} &= (I - A^{-1})P_0^x(I - A^{-1})^T + A^{-1}B_d P_0^d (A^{-1}B_d)^T + (I - A^{-1})P_0^{xd}(A^{-1}B_d)^T + \\ &\quad A^{-1}B_d(P_0^{xd})^T(I - A^{-1})^T \\ P_0^{12} &= (I - A^{-1})P_0^x C^T + A^{-1}B_d P_0^d C_d^T + (I - A^{-1})P_0^{xd}C_d^T + A^{-1}B_d(P_0^{xd})^T C^T \\ P_0^{22} &= CP_0^x C^T + C_d P_0^d C_d^T + CP_0^{xd}C_d^T + C_d(P_0^{xd})^T C^T \end{aligned} \tag{26}$$

in which:

$$P_0^x = E\{(x_0 - \hat{x}_0)(x_0 - \hat{x}_0)^T\} \qquad P_0^d = E\{(d_0 - \hat{d}_0)(d_0 - \hat{d}_0)^T\}$$
$$P_0^{xd} = E\{(x_0 - \hat{x}_0)(d_0 - \hat{d}_0)^T\}$$

3.2 Choice of the noise model

From eqn. 15 we can derive the corresponding velocity algorithm:

$$\begin{aligned} \Delta x_{k+1} &= A\Delta x_k + B\Delta u_k + B_d G_d w_{k-1}^d + G(w_k^x - w_{k-1}^x) \\ z_{k+1} &= CA\Delta x_k + z_k + CB\Delta u_k + CB_d G_d w_{k-1}^d + CG(w_k^x - w_{k-1}^x) + C_d G_d w_k^d \\ y_k - y_t &= z_k + v_k \end{aligned} \tag{27}$$

where y_t is the target. The previous equations can be written in the following augmented state-space:

$$\begin{aligned} \begin{bmatrix} \Delta x_{k+1} \\ z_{k+1} \end{bmatrix} &= \begin{bmatrix} A & 0 \\ CA & I \end{bmatrix} \begin{bmatrix} \Delta x_k \\ z_k \end{bmatrix} + \begin{bmatrix} B \\ CB \end{bmatrix} \Delta u_k + \begin{bmatrix} G & B_d G_d & 0 \\ CG & CB_d G_d & C_d G_d \end{bmatrix} \begin{bmatrix} w_k^x - w_{k-1}^x \\ w_{k-1}^d \\ w_k^d \end{bmatrix} \\ y_k - y_t &= \begin{bmatrix} 0 & I \end{bmatrix} \begin{bmatrix} \Delta x_k \\ z_k \end{bmatrix} + v_k \end{aligned} \tag{28}$$

Let

$$\tilde{G} = \begin{bmatrix} G & B_d G_d & 0 \\ CG & CB_d G_d & C_d G_d \end{bmatrix} \qquad \text{and} \qquad \tilde{w}_k = \begin{bmatrix} w_k^x - w_{k-1}^x \\ w_{k-1}^d \\ w_k^d \end{bmatrix} \qquad \tilde{v}_k = v_k \tag{29}$$

we obtain using eqn. 16:

$$Q'_{\tilde{w}} = E\{\tilde{w}_k \tilde{w}_k^T\} = \begin{bmatrix} 2Q_x & -Q_{xd} & Q_{xd} \\ -Q_{xd}^T & Q_d & 0 \\ Q_{xd}^T & 0 & Q_d \end{bmatrix} \qquad R_{\tilde{v}} = E\{\tilde{v}_k \tilde{v}_k^T\} = R_v \tag{30}$$

The previous equations contain terms that have meaning only at time $k > 0$; at time $k = 0$ we should modify the matrix $Q'_{\tilde{w}}$ as follows:

$$Q'_{\tilde{w}}(0) = \begin{bmatrix} Q_x & 0 & Q_{xd} \\ 0 & 0 & 0 \\ Q_{xd}^T & 0 & Q_d \end{bmatrix} \tag{31}$$

Thus, with the previous definitions we have obtained the general formulation of the velocity algorithm given in eqs. 9 and 10. Moreover, these relationships guarantee a correspondence between the obtained velocity algorithm and the given traditional algorithm.

3.3 The standard algorithm derived from the velocity algorithm

In this section we consider the velocity algorithm as given:

$$\begin{aligned} \begin{bmatrix} \Delta x_{k+1} \\ z_{k+1} \end{bmatrix} &= \begin{bmatrix} A & 0 \\ CA & I \end{bmatrix} \begin{bmatrix} \Delta x_k \\ z_k \end{bmatrix} + \begin{bmatrix} B \\ CB \end{bmatrix} \Delta u_k + \tilde{G}\tilde{w}_k \\ y_k - y_t &= \begin{bmatrix} 0 & I \end{bmatrix} \begin{bmatrix} \Delta x_k \\ z_k \end{bmatrix} + \tilde{v}_k \end{aligned} \tag{32}$$

in which $\tilde{w}_k$ and $\tilde{v}_k$ are random sequences with covariance matrices $Q'_{\tilde{w}}$ and $R_{\tilde{v}}$ given. Moreover, the initial estimate $\begin{bmatrix} \Delta\hat{x}_0 \\ \hat{z}_0 \end{bmatrix}$, and its covariance matrix, P_0 are given.

Eqs 14 and 12 clearly show that the covariance matrix evolution and the consequent gain of the Kalman filter depend on the product $\tilde{G}Q'_{\tilde{w}}\tilde{G}^T$ rather than the single matrices $\tilde{G}$ and $Q'_{\tilde{w}}$. Therefore, let

$$Q_{\tilde{w}} = \tilde{G}Q'_{\tilde{w}}\tilde{G}^T = \begin{bmatrix} Q_{11} & Q_{12} \\ (Q_{12})^T & Q_{22} \end{bmatrix} \tag{33}$$

where

$$\begin{aligned}
Q_{11} &= 2GQ_xG^T + B_dG_dQ_d(B_dG_d)^T - B_dG_dQ_{xd}^TG^T - GQ_{xd}(B_dG_d)^T \\
Q_{12} &= 2GQ_x(CG)^T + B_dG_dQ_d(CB_dG_d)^T - B_dG_dQ_{xd}^T(CG)^T \\
&\quad - GQ_{xd}(CB_dG_d)^T + GQ_{xd}(C_dG_d)^T \\
&= Q_{11}C^T + GQ_{xd}(C_dG_d)^T \\
Q_{22} &= 2CGQ_x(CG)^T + CB_dG_dQ_d(CB_dG_d)^T - CB_dG_dQ_{xd}^T(CG)^T \\
&\quad - CGQ_{xd}(CB_dG_d)^T + C_dG_dQ_{xd}^T(CG)^T \\
&\quad + CGQ_{xd}(C_dG_d)^T + C_dG_dQ_d(C_dG_d)^T \\
&= CQ_{11}C^T + C_dG_dQ_{xd}^T(CG)^T + CGQ_{xd}(C_dG_d)^T + C_dG_dQ_d(C_dG_d)^T
\end{aligned} \tag{34}$$

Combining eqs 24, 26, 30 and 34 we obtain the following nonlinear systems:

$$\begin{aligned}
\Delta\hat{x}_0 &= (I - A^{-1})\hat{x}_0 + A^{-1}B_d\hat{d}_0 - A^{-1}Bu_{-1} \\
\hat{z}_0 + y_t &= C\hat{x}_0 + C_d\hat{d}_0 \\
P_0^{11} &= (I - A^{-1})P_0^x(I - A^{-1})^T + A^{-1}B_dP_0^d(A^{-1}B_d)^T \\
&\quad + (I - A^{-1})P_0^{xd}(A^{-1}B_d)^TA^{-1}B_d(P_0^{xd})^T(I - A^{-1})^T \\
P_0^{12} &= (I - A^{-1})P_0^xC^T + A^{-1}B_dP_0^dC_d^T + (I - A^{-1})P_0^{xd}C_d^T \\
&\quad + A^{-1}B_d(P_0^{xd})^TC^T \\
P_0^{22} &= CP_0^xC^T + C_dP_0^dC_d^T + CP_0^{xd}C_d^T + C_d(P_0^{xd})^TC^T \\
Q_{11} &= 2GQ_xG^T + B_dG_dQ_d(B_dG_d)^T - B_dG_dQ_{xd}^TG^T - GQ_{xd}(B_dG_d)^T \\
Q_{12} - Q_{11}C^T &= CGQ_{xd}(CB_dG_d)^T \\
Q_{22} - CQ_{11}C^T &= C_dG_dQ_{xd}^T(CG)^T + CGQ_{xd}(C_dG_d)^T + C_dG_dQ_d(C_dG_d)^T \\
R_{\tilde{v}} &= R_v
\end{aligned} \tag{35}$$

in which the unknowns are $\hat{x}_0$, $\hat{d}_0$, B_d, C_d, P_0^x, P_0^d, P_0^{xd}, G, Q_x, G_d, Q_d, Q_{xd}, R_v. Moreover, the previous system must be solved with the following constraints:

$$\begin{bmatrix} P_0^x & P_0^{xd} \\ (P_0^{xd})^T & P_0^d \end{bmatrix} \geq 0 \qquad \begin{bmatrix} Q_x & Q_{xd} \\ (Q_{xd})^T & Q_d \end{bmatrix} \geq 0 \tag{36}$$

Let n be number of states and p be the number of the outputs; in general, the maximum dimension of an observable disturbance vector is p. In the following table a detail of the dimensions of each unknown is provided.

Unknown	Dimension	Properties	Ind. elements
$\hat{x}_0$	$n \times 1$		n
$\hat{d}_0$	$p \times 1$		p
B_d	$n \times p$		np
C_d	$p \times p$		p^2
P_0^x	$n \times n$	symm.	$n(n+1)/2$
P_0^d	$p \times p$	symm.	$p(p+1)/2$
P_0^{xd}	$n \times p$		np
G	$n \times n$		n^2
G_d	$p \times p$		p^2
Q_x	$n \times n$	symm.	$n(n+1)/2$
Q_d	$p \times p$	symm.	$p(p+1)/2$
Q_{xd}	$n \times p$		np
R_v	$p \times 1$		p

The number of independent equations of the system in eqn. 35 is given by the independent element of each Left Hand Side member, as reported in the next table.

Left Hand Side term	Dimension	Properties	Ind. equations
$\Delta\hat{x}_0$	$n \times 1$		n
$\hat{z}_0 + y_t$	$p \times 1$		p
P_0^{11}	$n \times n$	symm.	$n(n+1)/2$
P_0^{12}	$n \times p$		np
P_0^{22}	$p \times p$	symm.	$p(p+1)/2$
Q_{11}	$n \times n$	symm.	$n(n+1)/2$
$Q_{12} - Q_{11}C^T$	$n \times p$		np
$Q_{22} - CQ_{11}C^T$	$p \times p$	symm.	$p(p+1)/2$
$R_{\tilde{v}}$	$p \times 1$		p

Let α be the number of independent unknowns elements and β the number of independent equations. From the previous tables we have:

$$\alpha - \beta = 2p^2 + np + n^2 \tag{37}$$

which is greater or equal to 4. For several test problems this system has been solved using Nonlinear Programming routines (NPSOL or FMINCON).

4 Differences between the traditional and the velocity algorithm

In the previous sections a number of relations have been presented, which establish a correspondence between the traditional algorithm and the velocity formulation. However, in this section we present some control problems in which the two algorithms show a different behavior. These issues are:

- offset-free control in the presence of model-plant mismatch and/or unmodeled disturbances,
- inferential control,
- singular gain systems and non-square systems,
- unreachable targets.

4.1 Offset-free control

In the traditional state-space approach, in order to achieve offset-free control in the presence of model-plant mismatch and/or unmodeled disturbances it is common to add an integrated input or output disturbance model, whose estimate is updated by using the plant measurements. Both of these kinds of disturbance model are covered in eqn. 15. This "fictitious" disturbance is able to take into account the differences between the plant and the model and, therefore, it guarantees offset-free control. In particular, the disturbance estimate is used to compute the correct target for the state variables x_k, by assuming that the disturbance estimate remains constant in the future. When a disturbance model is not used, if the plant and the model do not agree, the target calculated for the states is wrong and the control algorithm shows steady-state offset.

On the other hand, a disturbance model does not appear in the velocity algorithm except in the initialization of the algorithm itself (see eqs. 24, 26) and in the matrix $Q_{\tilde{w}}$. However, offset-free control properties of this algorithm are guaranteed by the fact that the target of the variables Δx_k and z_k is always correct (*i.e.* the target is always zero) even if the plant and model do not agree.

Let us consider the standard algorithm without disturbance model:

$$\begin{aligned} x_{k+1} &= Ax_{k+1} + Bu_k + Gw_k^x \\ y_{k+1} &= Cx_k + v_k \end{aligned} \tag{38}$$

with initial estimate $\hat{x}_0$ and initial covariance matrix P_0^x given, in which w_k^x and v_k are random sequences with covariances Q_x and R_v, respectively. The corresponding velocity algorithm is (see eqs. 24, 26 and 29):

$$\begin{aligned} \begin{bmatrix} \Delta x_{k+1} \\ z_{k+1} \end{bmatrix} &= \begin{bmatrix} A & 0 \\ CA & I \end{bmatrix} \begin{bmatrix} \Delta x_k \\ z_k \end{bmatrix} + \begin{bmatrix} B \\ CB \end{bmatrix} \Delta u_k + \begin{bmatrix} G \\ CG \end{bmatrix} \tilde{w}_k \\ y_k - y^t &= \begin{bmatrix} 0 & I \end{bmatrix} \begin{bmatrix} \Delta x_k \\ z_k \end{bmatrix} + \tilde{v}_k \end{aligned} \tag{39}$$

where $\tilde{w}_k$ and $\tilde{v}_k$ are random sequences with covariance $2Q_x$ and R_v, respectively. These two algorithms have the same nominal response but the standard algorithm shows offset when there is mismatch between the plant and the model. This case is showed in the simulation section.

4.2 Inferential control

In several control problems the controlled variables are not measured outputs of the process. When a model of the process is available, estimates of the controlled variables can be obtained given the plant measurements by using a Kalman filter, provided that these controlled variables are detectable. This problem, known as inferential control, can be stated as follows.

Let $x \in \mathbb{R}^n$ be the state vector, $y^m \in \mathbb{R}^p$ the measured output vector and $y^c \in \mathbb{R}^q$ the vector of the controlled variable. For ease, we assume that all the controlled variables are not measured. A generic linear model for this process can be written as:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k \\ y_k^m &= Cx_k \\ y_k^c &= Hx_k \end{aligned} \tag{40}$$

From the LQ estimation theory, the state vector $x \in \mathbb{R}^n$ is detectable given the measurement vector $y^m \in \mathbb{R}^p$ if and only if the Hautus matrix has rank n, *i.e.* [8]:

$$\operatorname{rank} \mathcal{H} = \begin{bmatrix} \lambda I - A \\ C \end{bmatrix} = n \qquad \forall \lambda \in \mathbb{C} | \lambda | \geq 1 \tag{41}$$

For this kind of problem the corresponding velocity algorithm is:

$$\begin{aligned} \begin{bmatrix} \Delta x_{k+1} \\ z_{k+1}^m \\ z_{k+1}^c \end{bmatrix} &= \begin{bmatrix} A & 0 & 0 \\ CA & I & 0 \\ HA & 0 & I \end{bmatrix} \begin{bmatrix} \Delta x_k \\ z_k^m \\ z_k^c \end{bmatrix} + \begin{bmatrix} B \\ CB \\ HB \end{bmatrix} \Delta u_k \\ y_k^m &= \begin{bmatrix} 0 & I & 0 \end{bmatrix} \begin{bmatrix} \Delta x_k \\ z_k^m \\ z_k^c \end{bmatrix} + \tilde{v}_k \end{aligned} \tag{42}$$

For this system the Hautus test becomes:

$$\operatorname{rank} \begin{bmatrix} \lambda I - A & 0 & 0 & \\ -CA & (\lambda - 1) I & 0 & \\ -HA & 0 & (\lambda - 1) & 0 \\ 0 & I & 0 & \end{bmatrix} = n + p + q \qquad \forall \lambda : |\lambda| \geq 1 \tag{43}$$

Clearly, for $\lambda = 1$ the Hautus matrix has at most rank $n + p$. Therefore, the state vector of the velocity algorithm is not detectable and a stable estimator for this systems cannot be constructed.

Hence, the velocity algorithm is not able to address the problem of inferential control. There is an exception to this statement and it is when the controlled variables are a linear combination of the measured outputs, *i.e.* $z = Hy$.

4.3 Control of singular systems and systems with more outputs than inputs

We consider the case of square systems for which the gain matrix is singular. Given a generic state-space model, described by the following:

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k \\ y_k &= Cx_k \end{aligned} \tag{44}$$

the gain matrix is defined as:

$$K_p = C(I - A)^{-1}B \tag{45}$$

The gain matrix exists if and only if $(I - A)^{-1}$ is defined, *i.e.* if there are no integrating modes. Moreover, a process is singular if and only if $\det(K_p) = 0$. For such cases we have the following result.

Theorem 1 (Singular systems) *Given a process for which (A, B) is a controllable (stabilizable) pair, (C, A) is an observable (detectable) pair, and the gain matrix as defined in* (45) *is singular, using the state-space representation in* (2) *the pair $(\tilde{A}, \tilde{B})$ is not stabilizable.*

Proof. We prove this result by contradiction. Let assume that the pair $(\tilde{A}, \tilde{B})$ is stabilizable. The Hautus Lemma applied to (2) states:

$$\operatorname{rank} \mathcal{H} = \operatorname{rank} \begin{bmatrix} \lambda I - \tilde{A} & \tilde{B} \end{bmatrix} = n + p \qquad \forall \lambda(\tilde{A}) : |\lambda| \geq 1 \tag{46}$$

The eigenvalues of $\tilde{A}$ are the union of the eigenvalues of A and the eigenvalues of I. Therefore, the rank condition test must be checked at least for $\lambda = 1$. We have:

$$\operatorname{rank} \mathcal{H} = \begin{bmatrix} I - A & 0 & B \\ -CA & 0 & CB \end{bmatrix} = \operatorname{rank} \mathcal{H}_r = \begin{bmatrix} I - A & B \\ -CA & CB \end{bmatrix} = n + p \tag{47}$$

The matrix $\mathcal{H}_r$ is square and full rank, therefore invertible. Using the inversion matrix theorem, under the assumption that $(I - A)^{-1}$ exists, the inverse of H_r can be written as follows:

$$\mathcal{H}^{-1} = \begin{bmatrix} (I - A)^{-1}(I - B\Gamma^{-1}CA(I - A)^{-1}) & -(I - A)^{-1}B\Gamma^{-1} \\ -\Gamma^{-1}CA(I - A)^{-1} & \Gamma^{-1} \end{bmatrix} \tag{48}$$

in which the common term Γ^{-1} is given by:

$$\Gamma^{-1} = [CB + CA(I - A)^{-1}B]^{-1} = [C(I - A)^{-1}B]^{-1} = K_p^{-1} \tag{49}$$

which is in contradiction with the singularity of the process gain. □

For non-square systems with more controlled variables than manipulated variables we have the following result.

Theorem 2 *Given a linear system with m inputs and p outputs, with $p > m$, (A, B) stabilizable and (C, A) detectable, the corresponding velocity algorithm augmented system in (2) is not stabilizable.*

Proof. Let n be the number of states, from the Hautus lemma we have that the augmented system (2) is stabilizable if and only if

$$\operatorname{rank} \mathcal{H} = \begin{bmatrix} \lambda I - A & 0 & B \\ -CA & (\lambda - 1)I & CB \end{bmatrix} = n + p, \qquad \forall \lambda \in \mathbb{C} : |\lambda| \geq 1 \tag{50}$$

For $\lambda = 1$, we have:

$$\operatorname{rank} \mathcal{H} = \operatorname{rank} \begin{bmatrix} I - A & B \\ -CA & CB \end{bmatrix} \leq n + m < n + p \tag{51}$$

from which it follows immediately that the augmented system is not stabilizable. □

These results prevent us from using the velocity algorithm in infinite horizon MPC control schemes in the above mentioned cases. A finite horizon strategy can still be used, as discussed in the next section. It is interesting to point out that for such cases the standard algorithm can still be applied. The target calculation returns feasible targets for state and input, and the dynamic optimization computes the control moves to reach these feasible targets. In general, the reachable output target differs from the desired one, because of the singularity of K_p or because there are less manipulated inputs than controlled outputs.

4.4 Unreachable targets

In this section we consider the case where the desired output target cannot be reached. This possibility occurs when there are no feasible state and input vectors, x_s and u_s respectively, such that the following system has solution:

$$\begin{aligned} x_s &= Ax_s + Bu_s \\ \bar{y} &= Cx_s \end{aligned} \tag{52}$$

For instance, this is the case of singular system or non-square systems with more controlled outputs than manipulated inputs. Another possibility is that (52) does not admit solution due to input constraints:

$$Du_s \leq d \tag{53}$$

In all such cases, the deviation variable $z_k = y_k - \bar{y}$ never becomes zero at steady state and, clearly, an infinite horizon strategy cannot be implemented because any feasible input sequence would lead to an unbounded objective function. Therefore, a finite horizon approach must be chosen and, in general, stability of the MPC controller cannot be guaranteed. In fact, an equality constraint on the terminal state cannot be enforced because the corresponding quadratic program would be infeasible regardless the horizon length.

The finite horizon objective function can be written as follows:

$$\Phi = \sum_{k=0}^{N-1} [(y_k - \bar{y})^T Q (y_k - \bar{y}) + \Delta u_k^T S \Delta u_k] + \tilde{x}_N^T \Pi \tilde{x}_N \tag{54}$$

in which Π is the finite penalty matrix, which can be chosen to improve stability of the controller. At each sampling time the control sequence $\pi = [\Delta u_0, \Delta u_1, \ldots]^T$ is computed as solution of the following quadratic program:

$$\min_{\pi} \frac{1}{2} \pi^T H \pi + h^T \pi \tag{55}$$

subject to the following input constraint:

$$\Theta \pi \leq \theta \tag{56}$$

The matrix H and the vector h can be computed by expanding the objective function (54) and using the model equation (2) to replace the state variables in terms of the input. In particular, it can be shown that the linear term has the following form:

$$h = \alpha \Delta x_0 + \beta z_0 \tag{57}$$

in which $\alpha \in \mathbb{R}^{Nm \times n}$, $\beta \in \mathbb{R}^{Nm \times p}$ are appropriate matrices. The input constraint matrix and vector are reported below:

$$\Theta = \begin{bmatrix} I & 0 & \cdots & 0 \\ I & I & & \\ \vdots & & \ddots & \vdots \\ I & I & \cdots & I \\ -I & 0 & \cdots & 0 \\ -I & -I & & \\ \vdots & & \ddots & \vdots \\ -I & -I & \cdots & -I \end{bmatrix}; \qquad \theta = \begin{bmatrix} u_{\max} - u_{-1} \\ \\ \vdots \\ u_{\max} - u_{-1} \\ -u_{\min} + u_{-1} \\ \\ \vdots \\ -u_{\max} + u_{-1} \end{bmatrix} \tag{58}$$

in which u_{-1} is the previous injected input. In a receding horizon fashion, only the first input in injected into the plant and the remaining computed input increments are discarded. More precisely, the injected input is:

$$u_0 = u_{-1} + \Delta u_0 \tag{59}$$

This procedure is then repeated at the next sampling time.

In this section we emphasize two properties of the velocity algorithm, which occur when the desired output target $\bar{y}$ cannot be reached:

1. The computed sequence, π, does not become stationary, *i.e.* $\pi = 0$ for all elements.

2. The final steady-state point, *i.e.* x_s and u_s, depends on the horizon length.

The second property can be easily verified by the use of simulations and evaluating the final steady state for different horizon lengths. However, we can find the final state by solving the following nonlinear program:

$$\min_{x_s,u_s} |\Delta_0| \tag{60}$$

subject to:

$$\begin{aligned} x_s &= Ax_s + Bu_s \\ \pi &= \begin{bmatrix}\Delta_0 & \Delta_1 & \cdots\end{bmatrix}^T \text{ solution of the QP with} \\ \Delta x_0 &= 0; \qquad z_0 = Cx_s - \bar{y}; \qquad u_{-1} = u_s \end{aligned} \tag{61}$$

This problem returns the steady-state vectors (x_s, u_s) for which the initial control move of the corresponding quadratic program solution is zero. If $\Delta u_0 = 0$, the injected input is stationary $u_0 = u_{-1} = u_s$ and so is the state x_s because of (61). By solving this nonlinear program, we can verify that the solution, *i.e.* the steady state of the closed loop system, depends on the horizon length. Again, it is important to point out that the position algorithm finds the final steady-state by solving the target calculation problem, which is totally unrelated to the dynamic optimization and the horizon length. Using the velocity algorithm, the final steady-state is not known without solving a NLP as (60) (61), which is in general difficult to handle.

Now we consider the first property. Let x_s and u_s be the final steady state reached by applying the velocity algorithm, and let $Cx_s \neq \bar{y}$. We have the following result.

Theorem 3 (Stationarity of input sequence) *Let π be the solution of the quadratic program* (55) (56)*, with initial point $\Delta x_0 = 0$, $z_0 = Cx_s - \bar{y}$ and $u_{-1} = x_s$. Then, there are cases for which π is not stationary,* i.e. *$\pi = 0$ is not solution of this problem.*

Proof. We prove this result by contradiction. The necessary conditions for π to be solution of (55)(56), is that a vector $\lambda \in \mathbb{R}^{2mN}$ exists such that:

$$H\pi + \Theta^T\lambda + h = 0 \tag{62a}$$

$$-\Theta\pi + \theta \geq 0 \tag{62b}$$

$$\lambda \geq 0 \tag{62c}$$

$$\lambda_i(-\Theta\pi + \theta)_i = 0 \qquad i = 1, \ldots, 2mN \tag{62d}$$

Let assume $\pi = 0$ is solution of (62) and let $\lambda \geq 0$ be a vector of appropriate dimension. Clearly, (62b) and (62c) are satisfied. Θ and θ can be rearranged in such a way that first $n_1 \geq 0$ rows correspond to constraints active at steady state, *i.e.* either $(u_{\max} - u_s)_i = 0$ or $(-u_{\max} + u_s)_i = 0$:

$$\Theta = \begin{bmatrix}\Theta_1 \\ \Theta_2\end{bmatrix}; \qquad \theta = \begin{bmatrix}0 \\ \theta_2\end{bmatrix} \tag{63}$$

Denoting with λ_1 the vector formed by the first n_1 elements of λ and with λ_2 the vector formed by the remaining elements, from (62d) we have that $\lambda_2 = 0$. Hence (62a) becomes:

$$\Theta_1^T \lambda_1 + h = 0 \tag{64}$$

in which

$$h = \beta z_0 = \beta(Cx_s - \bar{y}) \neq 0 \tag{65}$$

It easy to construct cases for which all the elements of h are not zero. Hence, if the rank Θ_1 is less then the length of h, (64) does not admit solution, which is in contradiction with $\pi = 0$ solution of (55) (56). In particular, if no constraints are active at steady state (non-square or singular system case), if $\pi = 0$ from (62d) we have that $\lambda = 0$ and, therefore (62a) does not admit solution since $h \neq 0$. □

It interesting to point out that in the standard algorithm the solution is stationary, *i.e.* $\pi = \begin{bmatrix} u_0 - u_s & u_1 - u_s & \cdots \end{bmatrix} = 0$. In this algorithm, when the closed-loop system is at steady state, we have always $h = 0$ and, therefore, $\pi = 0$, $\lambda = 0$ is the optimal solution of the quadratic program. Thus, the open-loop and the closed-loop control sequence agree. Using the velocity algorithm, instead, the open-loop control sequence differs from the closed-loop receding horizon sequence, which is $\Delta u_0 = 0$.

5 Case studies

In this section we present two examples. The first one shows the effect of a correct and incorrect initialization of the velocity algorithm. It is also shown how the velocity algorithm achieves offset-free control in the presence of plant-model mismatch without an explicit disturbance model. The second example instead is used to show the behavior of the velocity algorithm when the set point is not reachable.

5.1 Example # 1

5.1.1 Plant and models

As example, we consider the following SISO plant:

$$\begin{aligned} x_{k+1} &= 0.5x_k + u_k \\ y_k &= x_k \end{aligned} \tag{66}$$

with initial state $x_0 = 0.5$. The initial estimate for the state is $\hat{x}_0 = 0.6$ with covariance $P_0^x = 0.01$. We assume the following model:

$$\begin{aligned} x_{k+1} &= 0.5x_k + u_k + w_k \\ y_k &= x_k + v_k \end{aligned} \tag{67}$$

with w_k and v_k random sequences with covariance $Q_x = 10^{-6}$ and $R_v = 10^{-6}$, respectively.

The corresponding model in the velocity form is:

$$\begin{bmatrix}\Delta x_{k+1}\\ z_{k+1}\end{bmatrix} = \begin{bmatrix}0.5 & 0\\ 0.5 & 1\end{bmatrix}\begin{bmatrix}\Delta x_k\\ z_k\end{bmatrix} + \begin{bmatrix}1\\ 1\end{bmatrix}\Delta u_k + \begin{bmatrix}1\\ 1\end{bmatrix}\tilde{w}_k$$
$$y_k - y_t = z_k + \tilde{v}_k \tag{68}$$

with $\tilde{w}_k$ and $\tilde{v}_k$ random sequences with covariance $Q_{\tilde{w}} = 2 \times 10^{-6}$ (at time $k = 0$ the covariance is $Q_{\tilde{w}} = 10^{-6}$) and $R_{\tilde{v}} = 10^{-6}$, respectively.

5.1.2 Controllers

For all controllers, the control objective function is:

$$\Phi(\hat{x}_{k|k}) = \sum_{j=k}^{\infty}(\hat{y}_{j|k})^2 + (\Delta u_j)^2 \tag{69}$$

subject to the model chosen (in the traditional or velocity position). Three different controllers are compared:

- Std: LQ controller based on the traditional model (eqn. 67);
- VA1: LQ controller based on the velocity algorithm model (eqn. 68), with correct initialization (see next section);
- VA2: LQ controller based on the velocity algorithm model (eqn. 68), with wrong initialization (see next section).

5.1.3 Initialization of the velocity algorithm

The correct initialization of the algorithm, according to eqs 24 and 26, is

$$\Delta\hat{x}_0 = -0.5 \qquad \hat{z}_0 = 0.5 \qquad P_0 = \begin{bmatrix}0.01 & -0.01\\ -0.01 & 0.01\end{bmatrix} \tag{70}$$

These are the value used by controller VA1. On the other hand, in order to emphasize the effect of a wrong initialization of the velocity algorithm, controller VA2 uses the following initial condition:

$$\Delta\hat{x}_0 = 0 \qquad \hat{z}_0 = 0.5 \qquad P_0 = \begin{bmatrix}0.01 & -0.01\\ -0.01 & 0.01\end{bmatrix} \tag{71}$$

5.1.4 Simulation

A simulation is presented here. First each controller is required to track the output from the initial state to the origin (target). Then, at time $t_d = 10$ an input disturbance enters the plant:

$$x_{k+1} = 0.5x_k + u_k + 0.5$$
$$y_k = x_k \tag{72}$$

and each controller is required to reject the disturbance. The simulation responses are reported in Figure 1. Some comments are appropriate:

1. when there is no disturbance, the standard algorithm and the velocity algorithm have the same response;

2. when the disturbance enters the plant, the traditional algorithm shows offset, while the velocity algorithm is able to reject the disturbance;

3. the velocity algorithm with wrong initial estimate for Δx_0 has a bad initial behavior but, when the disturbance enters the plant, it is able to reject it. The initialization problem has an effect only in the first part of the simulation until the system reaches the origin.

5.2 Example # 2

We consider the following non-square system with two outputs and one input:

$$A = \begin{bmatrix} 0.9 & 0 \\ 0 & 0.8 \end{bmatrix}, \qquad B = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \qquad C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{73}$$

MPC controllers with different finite horizon are compared. For each controller, the penalty matrices in eqn. 54 are

$$Q = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \qquad S = 1, \qquad \Pi = Q \tag{74}$$

It is interesting to note that the final penalty matrix cannot be chosen as the solution of the Riccati equation since the augmented system is not stabilizable.

We attempt to move both outputs from 0 to 1 but, since there is only one input, we expect to have offset in both variables. The results of the simulations are reported in Figure 2. As expected the steady-state value of the closed-loop outputs depends on the horizon It can be seen that when horizon increases the steady-state outputs approach their desired targets in a least-square sense, *i.e.* the value that a standard MPC controller with target calculation reaches (also shown).

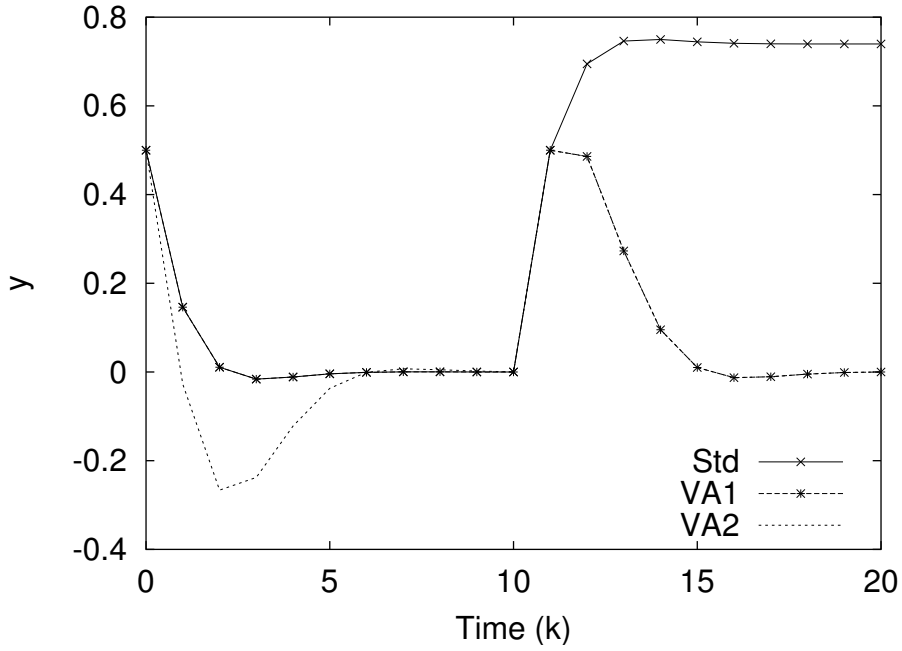
6 Observations and conclusions

In this paper a survey on the velocity algorithm has been presented in order to understand the theoretical relationship between this model description and the standard state-space approach.

Below are reported main advantages and disadvantages of this algorithm respect to the traditional formulation:

- Advantages of the velocity form

(a)



(b)

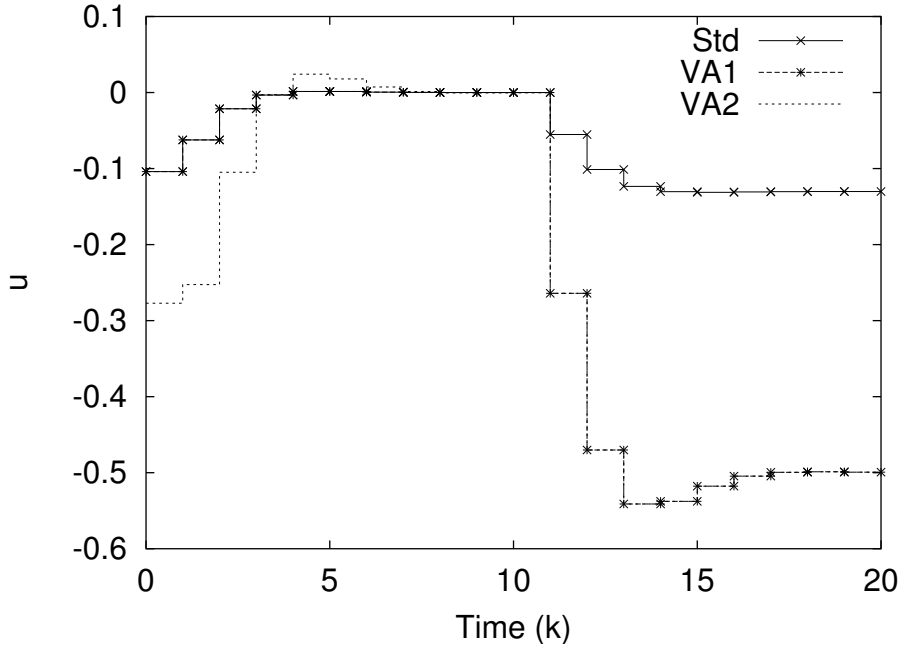


Figure 1: Output (a) and Input (b) responses for each controller (Example # 1)

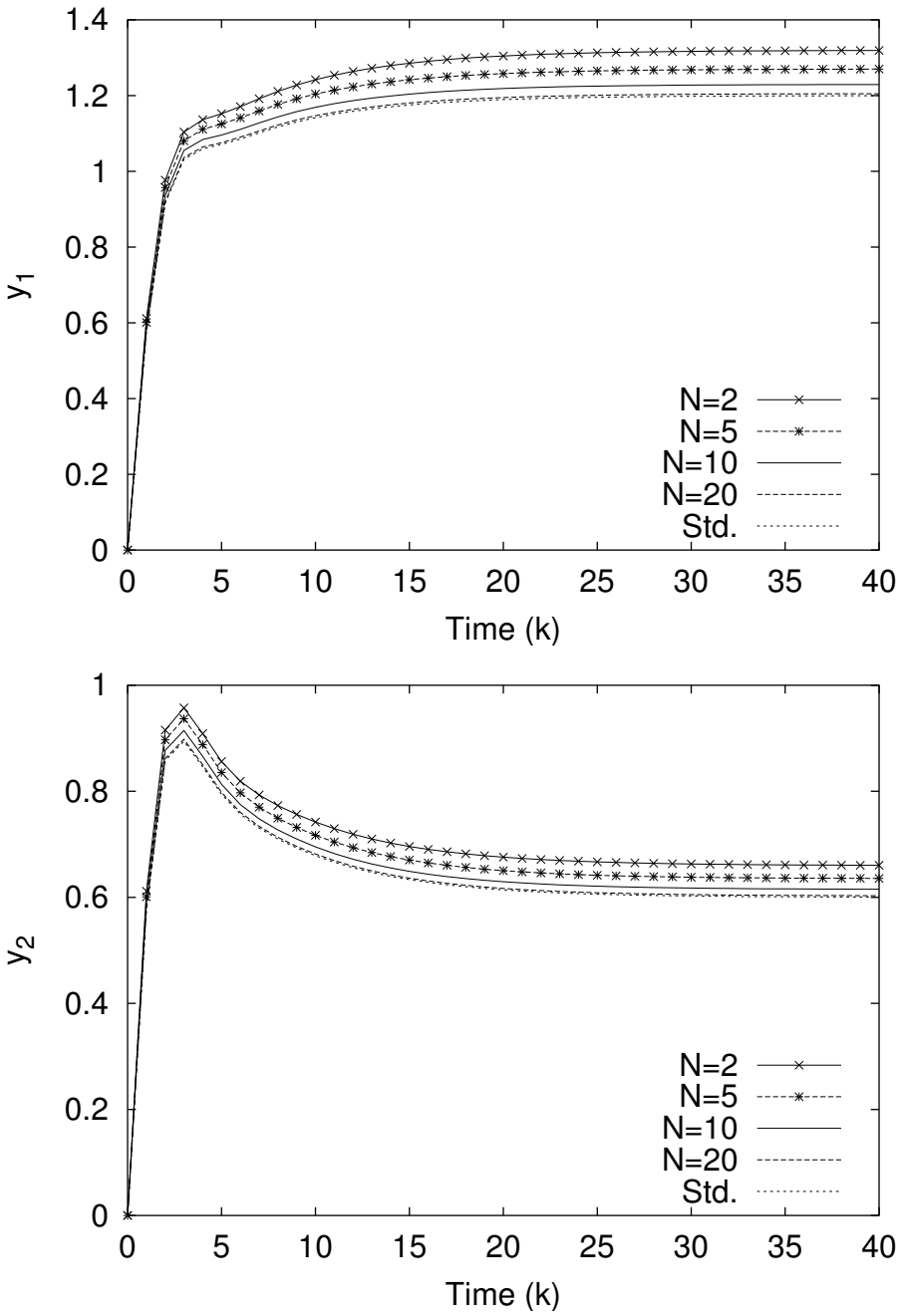


Figure 2: Outputs for velocity MPC controllers with different horizon and MPC with target calculation .(Example # 2)

1. It does not require the choice of a disturbance model: offset-free control is always achieved.
2. The choice of the disturbance model affects only the initial estimation of the state vector $\tilde{x}_0$ and the noise matrix $\tilde{G}$, but it does not affect the control law.
3. A target calculation is not required because the target for each state is always zero.

- Disadvantages of the velocity form

1. If the gain matrix of the system (in case of square system systems) is singular or there are more controlled variables than manipulated variables (non-square systems), the augmented system in velocity form is not stabilizable and infinite horizon cannot be used. In both cases one can still use a finite horizon, but this does not guarantee nominal stability. A terminal equality constraint cannot be enforced since the output target cannot be reached in such cases.
2. When the output target is unreachable, the steady state of the closed loop depends on the choice of the finite horizon.
3. It is not able to control variables that are not measured (inferential control) because the controlled variables are not observable in the velocity formulation. However, if the controlled variables are a linear combination of the measured outputs, the velocity form can still be applied.
4. The initialization of the state vector $\tilde{x}_0$, in terms of estimate $\hat{\tilde{x}}$ and covariance matrix P_0, is not obvious.
5. The choice of the noise model, in terms of $\tilde{G}$ and $Q_{\tilde{w}}$, is not obvious.
6. Since the state is augmented with the output, the MPC optimization problem may be more expensive to solve. This problem is common to any formulation that requires augmentation (for instance, the standard formulation requires augmentation if Δu penalties or constraints need to be considered).
7. The control law cannot include an input penalty term, *i.e.* $(u_k - u_s)^T R(u_k - u_s)$ because the target for the input, *i.e.* u_s cannot be specified in the velocity algorithm. Moreover, if we wanted to include this term in the LQR objective function (see eqn. 4) we would have to add u_{k-1} as additional state, but the obtained augmented state-space ($[\Delta x_k^T, z_k^T, u_{k-1}^T]^T$ and related matrices $\tilde{A}$ and $\tilde{B}$) would not be stabilizable.

References

[1] C. R. Cutler and B. L. Ramaker. Dynamic matrix control – a computer algorithm. In *AIChE 86th National Meeting*, Houston, TX, 1979.

[2] C. E. Garcia and A. M. Morshedi. Quadratic programming solution of dynamic matrix control (QDMC). *Chem. Eng. Commun.*, 46:73–87, 1986.

[3] D. E. Kassmann, T. A. Badgwell, and R. B. Hawkins. Robust steady-state target calculation for model predictive control. *AIChE J.*, 46:1007–1024, 2000.

[4] J. H. Lee, M. S. Gelormino, and M. Morari. Model predictive control of multi-rate sampled-data systems: a state-space approach. *Int. J. Control*, 55:153–191, 1992.

[5] B. A. Ogunnaike and W. H. Ray. *Process Dynamics, Modeling, and Control*. Oxford University Press, 1994.

[6] G. H. C. Oliveira, W. C. Amaral, G. Favier, and G. A. Dumont. Constrained robust predictive controller for uncertain processes modeled by orthonormal series functions. *Automatica*, 36:563–571, 2000.

[7] J. Richalet, J. Rault, J. L. Testud, and J. Papon. Model predictive heuristic control: Applications to industrial processes. *Automatica*, 14:413–1554, 1978.

[8] E. D. Sontag. *Mathematical Control Theory*. Springer, second edition, 1998.