6 Model-based predictive control

6.1 Introduction

Model-based predictive control (MPC) has become the most popular advanced control technology in the chemical processing industries. There are many variants of MPC controllers, both in academia and in industry, but they all share the common trait that an explicitly formulated process model is used to predict and optimize future process behaviour. Most MPC controllers are able to account for constraints both in manipulated variables and states/controlled variables through the formulation of the optimization problem.

When formulating the optimization problem in MPC, it is important to ensure that it can be solved in the short time available (i.e., the sampling interval is an upper bound on the acceptable time for performing the calculations). For that reason, the optimization problem is typically cast into one of two standard forms:

- Linear programming (LP) formulation. In an LP formulation, both the objective function and the constraints are linear.

- Quadratic programming (QP) formulation. In a QP formulation, the objective function is quadratic, whereas the constraints have to be linear. In addition, to ensure that there exists a unique optimal solution that can be found quickly with effective optimization solvers, the *Hessian matrix* in the objective function has to be *positive definite*[15].

LP problems can be solved more efficiently than QP problems, and an LP formulation may therefore be advantageous for very large optimization problems. However, a QP formulation generally leads to smoother control action and more intuitive effects of changes in the tuning parameters. The connection to 'traditional advanced control', i.e., linear quadratic (LQ) optimal control, is also much closer for a QP formulation than for an LP formulation. For these reasons, we will focus on a QP formulation in the following, and describe in some detail how a QP optimization problem in MPC may be formulated.

[15]The Hessian matrix defines the quadratic term in the objective function, and is a symmetric matrix. Positive definiteness means that all eigenvalues are positive - for a monovariable optimization problem this implies that the coefficient for the quadratic term in the objective function is positive.

6.2 Formulation of a QP problem for MPC

A standard QP problem takes the form

$$\min_{v} \quad 0.5v^T \widetilde{H} v + c^T v \tag{27}$$

subject to the constraints

$$Lv \leq b \tag{28}$$

Here v is the vector of free variables in the optimization, whereas $\widetilde{H}$ is the *Hessian matrix*, that was mentioned above, and which has to be positive definite. The vector c describes the linear part of the objective function, whereas the matrix L and the vector b describe the linear constraints. Some QP solvers allow the user to specify separate upper and lower bounds for v, whereas other solvers require such constraints to be included in L and b. For completeness, we will assume that these constraints have to be included in L and b.

The formulation of the MPC problem starts from a linear, *discrete-time* state-space model of the type

$$x_{k+1} = Ax_k + Bu_k \tag{29}$$
$$y_k = Cx_k \tag{30}$$

where the subscripts refer to the sampling instants. That is, subscript $k+1$ refers to the sample instant one sample interval after sample k. Note that for discrete time models used in control, there is normally no direct feed-through term, the measurement y_k does not depend on the input at time k, but it does depend on the input at time $k-1$ through the state x_k. The reason for the absence of direct feed-through is that normally the output is measured at time k before the new input at time k is computed and implemented.

In the same way as is common in control literature, the state x, input u and measurement y above should be interpreted as *deviation variables.* This means that they represent the deviations from some consistent set of of variables $\{x_L, u_L, y_L\}$ around which the model is obtained[16]. For a stable

[16]We do not here specify *how* the model is obtained, but typically it is either the result of identification experiments performed around the values $\{x_L, u_L, y_L\}$ or the result of linearizing and discretizing a non-linear, physical model around the values $\{x_L, u_L, y_L\}$.

process, the set $\{x_L, u_L, y_L\}$ will typically represent a steady state - often the steady state we want to keep the process at. To illustrate, if in y_L represents a temperature of $330K$, a physical measurement of $331K$ corresponds to a devation variable $y = 1K$.

A typical optimization problem in MPC might take the form

$$\begin{aligned} \min_u \quad f(x,u) &= \sum_{i=0}^{n-1} \{(x_i - x_{ref,i})^T Q (x_i - x_{ref,i}) \\ &\quad + (u_i - u_{ref,i})^T P (u_i - u_{ref,i})^T\} \\ &\quad + (x_n - x_{ref,n})^T S (x_n - x_{ref,n}) \end{aligned} \tag{31}$$

subject to constraints

$$\begin{aligned} x_0 &= \text{given} \\ U_L &\leq u_i \leq U_U \quad \text{for } 0 \leq i \leq n-1 \\ Y_L &\leq Hx_i \leq Y_U \quad \text{for } 1 \leq i \leq n+j \end{aligned} \tag{32}$$

In the objective function Eq. (31) above, we penalize the deviation of the states x_i from some desired reference trajectory $x_{ref,i}$ and the deviation of the inputs u_i from some desired trajectory $u_{ref,i}$. These reference trajectories are assumed to be given to the MPC controller by some outside source. They may be constant or may also vary with time (subscript i). The constraints on achievable inputs or acceptable states are usually not dependent on the reference trajectories, and therefore these reference trajectories do not appear in the constraint equations (32). Usually, the state constraints represent constraints on process measurements (giving $H = C$), but constraints on other combinations of states are also possible (including constraints on combinations of inputs and states).

In the following, this formulation of the optimization problem will be recast into the standard QP formulation in Eqs.(27) and (28), but first a number of remarks and explanations to the optimization problem formulation in Eqs.(31) to (32) are needed.

- In addition to the above constraints, it is naturally assumed that the process follows the model in Eqs. (29) and (30).
- The matrices Q, P, and S are all assumed to be symmetric. P and S are assumed to be positive definite, whereas Q may be positive semi-definite.

- In many applications it may be more natural to put a weight (or cost) on the actual measurements rather than the states. This can easily be done by choosing $Q = C^T\widetilde{Q}C$, where $\widetilde{Q}$ is the weight on the measurements.

- One may also put constraints on the rate of change of the inputs, giving additional constraints on the form $\Delta U_L \leq u_i - u_{i-1} \leq \Delta U_U$.

- For the output constraints in Eq. (32) to be well defined, we must specify how the inputs u_i should behave in the interval $n \leq i \leq n+j-1$. Typical choices for this time interval are either that $u_i = u_{ref,i}$ or that $(u_i - u_{ref,i}) = K(x_i - x_{ref,i})$. The latter choice assumes that a (stabilizing) state feedback controller is used in this time interval. Note that this controller will never be used in practice (since the MPC calculations are re-computed at each sample instant), but it is needed to make the constraints well defined.

- If one assumes that $(u_i - u_{ref,i}) = K(x_i - x_{ref,i})$ for $n \leq i \leq n+j-1$, one should also include the input constraints in the problem formulation for the time interval $n \leq i \leq n+j-1$. These input constraints then effectively become state constraints for this time interval.

- Some MPC formulations use an objective function of the form $f(x,u) = \sum_{i=0}^{n_p}(x_i - x_{ref,i})^T Q(x_i - x_{ref,i}) + \sum_{i=0}^{n_u}(u_i - u_{ref,i})^T P(u_i - u_{ref,i})$, where $n_p > n_u$, and typically assume that $u_i = u_{ref,i}$ for $n_u < i < n_p$. Note that this corresponds to a particular choice for 'terminal state weight' S, since x_i for $n_u + 1 < i \leq n_p$ will then be given by x_{n_u+1} (and the process model).

- It is common to introduce integral action in MPC controllers by using the input *changes* at time i as free variables in the optimization, rather than the input itself. This follows, since the actual inputs are obtained by integrating the changes in the input. This can be done within the same framework and model structure as above, using the model

$$\begin{aligned}
\widetilde{x}_{k+1} &= \begin{bmatrix} x_{k+1} \\ u_k \end{bmatrix} = \widetilde{A}\widetilde{x}_k + \widetilde{B}\Delta u_k \\
y_k &= \widetilde{C}\widetilde{x}_k
\end{aligned}$$

where $\Delta u_k = u_k - u_{k-1}$, and

$$\widetilde{A} = \begin{bmatrix} A & B \\ 0 & I \end{bmatrix}, \quad \widetilde{B} = \begin{bmatrix} 0 \\ I \end{bmatrix}, \quad \widetilde{C} = \begin{bmatrix} C & 0 \end{bmatrix}$$

- To have a stable closed-loop system, it is necessary to have at least as many feedback paths as integrators, i.e., one needs at least as many (independent) measurements as inputs. When the number of inputs exceeds the number of measurements, it is common to define 'ideal resting values' for some inputs. This essentially involves putting some inputs in the measurement vector, and defining setpoints for these.

In the following, we will recast the MPC optimization problem as a standard QP problem. We will assume that $u_i - u_{ref,n} = K(x_i - x_{ref,n})$ for $n \leq i \leq n + j - 1$. To start off, we stack the state refernces $x_{ref,i}$, input references $u_{ref,i}$, input deviations $v_i = u_i - u_{ref,i}$ and state deviations $\chi_i = x_i - x_{ref,i}$ in long (column) vectors x_{ref}, u_{ref}, v and χ_{dev} :

$$\begin{aligned} u_{ref} &= \begin{bmatrix} u_{ref,0} \\ u_{ref,1} \\ \vdots \\ u_{ref,n-2} \\ u_{ref,n-1} \end{bmatrix}; \quad x_{ref} = \begin{bmatrix} x_{ref,1} \\ x_{ref,2} \\ \vdots \\ x_{ref,n-1} \\ x_{ref,n} \end{bmatrix}; \\ v &= \begin{bmatrix} v_0 \\ v_1 \\ \vdots \\ v_{n-2} \\ v_{n-1} \end{bmatrix}; \quad \chi_{dev} = \begin{bmatrix} \chi_1 \\ \chi_2 \\ \vdots \\ \chi_{n-1} \\ \chi_n \end{bmatrix} \end{aligned}$$

We will use the *superposition principle,* which states that the total effect of several inputs can be obtained simply by summing the effects of the individual inputs. The superposition principle is always valid for linear systems, but typically does not hold for non-linear systems. This allows us to first calculate the deviation from the desired state trajectory that would result, given the initial state x_0 and assuming that the nominal reference input u_{ref} is followed. This results in the trajectory of state deviations χ_{dev}.

Repeated use of the model equation Eq. (29) then gives

$$\begin{aligned}
\chi_{dev} &= \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^{n-1} \\ A^n \end{bmatrix} x_0 + \begin{bmatrix} B & 0 & \cdots & 0 & 0 \\ AB & B & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & 0 & 0 \\ A^{n-2}B & A^{n-3}B & \cdots & B & 0 \\ A^{n-1}B & A^{n-2}B & \cdots & AB & B \end{bmatrix} u_{ref} - x_{ref} \\
&= \widehat{A}x_0 + \widehat{B}u_{ref} - x_{ref}
\end{aligned}$$

Thus, we have obtained a deviation from the desired state trajectory x_{ref}, which should be counteracted using deviations $v_i = u_i - u_{ref,i}$ from the nominal input trajectory. Note that χ_{dev} is independent from the deviation from the deviation from the input reference trajectory, i.e., independent of v, and may therefore be calculated prior to solving the MPC optimization. Similarly, the effect of the deviations from the nominal input trajectory on the states is given by $\chi_v = \widehat{B}v$ (which clearly does depend on the result of the MPC optimization).

Introducing the matrices

$$\widehat{Q} = \begin{bmatrix} Q & 0 & \cdots & 0 & 0 \\ 0 & Q & \ddots & \vdots & \vdots \\ 0 & 0 & \ddots & 0 & 0 \\ \vdots & \vdots & \ddots & Q & 0 \\ 0 & 0 & \cdots & 0 & S \end{bmatrix}, \quad \widehat{P} = \begin{bmatrix} P & 0 & \cdots & 0 & 0 \\ 0 & P & \ddots & \vdots & \vdots \\ 0 & 0 & \ddots & 0 & 0 \\ \vdots & \vdots & \ddots & P & 0 \\ 0 & 0 & \cdots & 0 & P \end{bmatrix} \tag{33}$$

the objective function can be written as

$$\begin{aligned}
f(x,u) = f(\chi_{dev}, \chi_v, v) &= (x_0 - x_{ref,0})^T Q (x_o + x_{ref,0}) + \\
&\qquad (\chi_{dev} + \chi_v)^T \widehat{Q} (\chi_{dev} + \chi_v) + v^T \widehat{P} v \\
&= (x_0 - x_{ref,0})^T Q (x_o + x_{ref,0}) + \chi_{dev}^T \widehat{Q} \chi_{dev} + \\
&\qquad 2\chi_{dev}^T \widehat{Q} \chi_v + \chi_v^T \widehat{Q} \chi_v + v^T \widehat{P} v
\end{aligned}$$

which should be minimized using the vector v as free variables.

Now, the terms $(x_0 - x_{ref,0})^T Q (x_o + x_{ref,0}) + \chi_{dev}^T \widehat{Q} \chi_{dev}$ will not be affected by the optimization, and may therefore be removed from the objective function.. This is because we are primarily interested in finding the inputs that minimize the objective function, and not in the optimal value of the objective function. Thus, the objective function is in the form of a standard QP problem as defined in Eqs. (27) and (28) if we define

$$\begin{aligned} \widetilde{H} &= \widehat{B}^T \widehat{Q} \widehat{B} + \widehat{P} \\ c^T &= \chi_{dev}^T \widehat{A}^T \widehat{Q} \widehat{B} \end{aligned} \tag{34}$$

It now remains to express the constraints in the MPC problem in the form of a standard QP problem. The lower and upper constraints on the manipulated variable from $0 \leq i \leq n-1$ simply become

$$Iv \geq \begin{bmatrix} U_L \\ \vdots \\ U_L \end{bmatrix} - u_{ref} \tag{35}$$

$$-Iv \geq -\begin{bmatrix} U_U \\ \vdots \\ U_U \end{bmatrix} + u_{ref} \tag{36}$$

Similarly, the constraints on the measurements/states for $1 \leq i \leq n$ become

$$\begin{bmatrix} H & 0 & \cdots & \cdots & 0 \\ 0 & H & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & H & 0 \\ 0 & \cdots & \cdots & 0 & H \end{bmatrix} \chi_v \quad \geq$$

$$\begin{bmatrix} Y_L \\ \vdots \\ \vdots \\ \vdots \\ Y_L \end{bmatrix} - \begin{bmatrix} H & 0 & \cdots & \cdots & 0 \\ 0 & H & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & H & 0 \\ 0 & \cdots & \cdots & 0 & H \end{bmatrix} \qquad (\chi_{dev} + x_{ref})$$

$$- \begin{bmatrix} H & 0 & \cdots & \cdots & 0 \\ 0 & H & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & H & 0 \\ 0 & \cdots & \cdots & 0 & H \end{bmatrix} \chi_v \quad \geq$$

$$- \begin{bmatrix} Y_U \\ \vdots \\ \vdots \\ \vdots \\ Y_U \end{bmatrix} + \begin{bmatrix} H & 0 & \cdots & \cdots & 0 \\ 0 & H & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & H & 0 \\ 0 & \cdots & \cdots & 0 & H \end{bmatrix} \qquad (\chi_{dev} + x_{ref})$$

$$\Updownarrow$$

$$\begin{bmatrix} H & 0 & \cdots & \cdots & 0 \\ 0 & H & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & H & 0 \\ 0 & \cdots & \cdots & 0 & H \end{bmatrix} \widehat{B} v \geq \begin{bmatrix} Y_L \\ \vdots \\ \vdots \\ \vdots \\ Y_L \end{bmatrix} - \begin{bmatrix} H & 0 & \cdots & \cdots & 0 \\ 0 & H & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & H & 0 \\ 0 & \cdots & \cdots & 0 & H \end{bmatrix} (\chi_{dev} + x_{ref}) \tag{37}$$

$$- \begin{bmatrix} H & 0 & \cdots & \cdots & 0 \\ 0 & H & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & H & 0 \\ 0 & \cdots & \cdots & 0 & H \end{bmatrix} \widehat{B} v \geq - \begin{bmatrix} Y_U \\ \vdots \\ \vdots \\ \vdots \\ Y_U \end{bmatrix} + \begin{bmatrix} H & 0 & \cdots & \cdots & 0 \\ 0 & H & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & H & 0 \\ 0 & \cdots & \cdots & 0 & H \end{bmatrix} (\chi_{dev} + x_{ref}) \tag{38}$$

We found above that

$$\begin{aligned} x_n &= A^n x_0 + \begin{bmatrix} A^{n-1}B & A^{n-2}B & \cdots & AB & B \end{bmatrix} (u_{ref} + v) \\ &= \chi_{dev,n} + x_{ref,n} + \begin{bmatrix} A^{n-1}B & A^{n-2}B & \cdots & AB & B \end{bmatrix} v \end{aligned}$$

The process model and the assumed control action $(u_i - u_{ref,n}) = K(x_i - x_{ref,n})$ for the time period $n \leq i \leq n + j - 1$ gives, after trivial, but tedious manipulation

$$\begin{aligned} x_i &= (A+BK)^{i-n}x_n + \left\{ \sum_{j=0}^{i-n-1} (A+BK)^i \right\} B(u_{ref,n} - Kx_{ref,n}) \\ u_i - u_{ref,n} &= K(x_i - x_{ref,n}) = K(A+BK)^{i-n}x_n \\ &\quad +K\left[\left\{ \sum_{k=0}^{i-n-1} (A+BK)^i \right\} B(u_{ref,n} - Kx_{ref,n}) - x_{ref,n} \right] \end{aligned}$$

which combined with the above expression for x_n results in

$$\begin{aligned} &\begin{bmatrix} K \\ K(A+BK)^1 \\ \vdots \\ K(A+BK)^{j-1} \\ K(A+BK)^j \end{bmatrix} \begin{bmatrix} A^{n-1}B & A^{n-2}B & \cdots & AB & B \end{bmatrix} v \geq \qquad (39) \\ &\qquad \begin{bmatrix} U_L \\ U_L \\ \vdots \\ U_L \\ U_L \end{bmatrix} - \begin{bmatrix} K \\ K(A+BK)^1 \\ \vdots \\ K(A+BK)^{j-1} \\ K(A+BK)^j \end{bmatrix} (\chi_{dev,n} + \quad x_{ref,n}) \\ &- \left\{ \begin{bmatrix} I \\ I \\ \vdots \\ I \\ I \end{bmatrix} + \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ I & 0 & \ddots & \vdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ I & \vdots & \ddots & 0 & 0 \\ I & I & \cdots & I & 0 \end{bmatrix} \begin{bmatrix} K \\ K(A+BK)^1 \\ \vdots \\ K(A+BK)^{j-1} \\ K(A+BK)^j \end{bmatrix} B \right\} \\ &\qquad \times \begin{bmatrix} I & -K \end{bmatrix} \begin{bmatrix} u_{ref,n} \\ x_{ref,n} \end{bmatrix} \end{aligned}$$

$$-\begin{bmatrix} K \\ K(A+BK)^1 \\ \vdots \\ K(A+BK)^{j-1} \\ K(A+BK)^j \end{bmatrix} \begin{bmatrix} A^{n-1}B & A^{n-2}B & \cdots & AB & B \end{bmatrix} v \geq \tag{40}$$

$$-\begin{bmatrix} U_U \\ U_U \\ \vdots \\ U_U \\ U_U \end{bmatrix} + \begin{bmatrix} K \\ K(A+BK)^1 \\ \vdots \\ K(A+BK)^{j-1} \\ K(A+BK)^j \end{bmatrix} (\chi_{dev,n} + \quad x_{ref,n})$$

$$+\left\{ \begin{bmatrix} I \\ I \\ \vdots \\ I \\ I \end{bmatrix} + \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ I & 0 & \ddots & \vdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ I & \vdots & \ddots & 0 & 0 \\ I & I & \cdots & I & 0 \end{bmatrix} \begin{bmatrix} K \\ K(A+BK)^1 \\ \vdots \\ K(A+BK)^{j-1} \\ K(A+BK)^j \end{bmatrix} B \right\} \times$$

$$\begin{bmatrix} I & -K \end{bmatrix} \begin{bmatrix} u_{ref,n} \\ x_{ref,n} \end{bmatrix}$$

$$\begin{bmatrix} H(A+BK) \\ H(A+BK)^2 \\ \vdots \\ H(A+BK)^{j-1} \\ H(A+BK)^j \end{bmatrix} \begin{bmatrix} A^{n-1}B & A^{n-2}B & \cdots & AB & B \end{bmatrix} v \geq \tag{41}$$

$$\begin{bmatrix} Y_L \\ Y_L \\ \vdots \\ Y_L \\ Y_L \end{bmatrix} - \begin{bmatrix} H(A+BK) \\ H(A+BK)^2 \\ \vdots \\ H(A+BK)^{j-1} \\ H(A+BK)^j \end{bmatrix} (\chi_{dev,n} + x_{ref,n})$$

$$-\begin{bmatrix} H(A+BK) \\ H(A+BK)^2 \\ \vdots \\ H(A+BK)^{j-1} \\ H(A+BK)^j \end{bmatrix} \begin{bmatrix} A^{n-1}B & A^{n-2}B & \cdots & AB & B \end{bmatrix} v \geq \tag{42}$$

$$-\begin{bmatrix} Y_U \\ Y_U \\ \vdots \\ Y_U \\ Y_U \end{bmatrix} + \begin{bmatrix} H(A+BK) \\ H(A+BK)^2 \\ \vdots \\ H(A+BK)^{j-1} \\ H(A+BK)^j \end{bmatrix} (\chi_{dev,n} + x_{ref,n})$$

The overall set of constraints for the MPC problem is now obtained by simply stacking equations (35,36,37,38,39,40,41,42). All these constraint equations have a left hand side consisting of a matrix multiplied with the vector of free variables in the optimization, and a right hand side which is vector-valued (and which can be evaluated prior to the optimization. Note that the introduction of non-zero (and possibly time-varying) reference trajectories significantly complicate the expressions, in particular for the constraints in the period $n \leq i \leq n+j$.

There is a slight difference between the state constraint equations and the input constraint equations, in that Eqs. (35) and (36) include an input constraint at time zero (the present time), whereas the state constraint equations (Eqs. (37) and (38)) do not. This is because the state constraints cannot be enforced if they are violated at time zero, since the present state is unaffected by present and future inputs. Note also that Eqs. (39) and (40) covers the input constraints from time n until $n+j$, whereas Eqs. (41) and (42) covers the state constraints from time $n+1$ unti $n+j$. The state constraints for time n is covered by Eqs. (37) and (38).

6.3 Step response models

In industrial practice, process models based on step response descriptions have been very successful. Whereas step response models have no theoretical advantages, they have the practical advantage of being easier to understand for engineers with little background in control theory.

With a soild understanding of the material presented above, the capable reader should have no particular problem in developing a similar MPC formulation based on a step response model. Descriptions of such formulations can also be found in available publications, like Garcia and Morshedi's [16] original paper presenting "Quadratic Dynamic Matrix Control". Alternatively, step response models may also be expressed in state space form (with a larger number of states than would be necessary in a "minimal" state space model), see e.g. [28] for details.

The reader should beware that step-response models have "finite memory", and hence should only be used for asymptotically stable processes, that is, processes where the effect of old inputs vanish over time. Most industrially successful MPC controllers based on step response models are modified to handle also integrating processes, whereas truly unstable processes cannot be handled. Handling unstable processes using step response models would require more complex modifications to the controllers and model description, and would thereby remove the step response model's advantage of being easy to understand.

Partly due to these reasons, MPC controllers are seldom used on unstable processes. If the underlying process is unstable, it is usually first stabilised by some control loops, and the MPC controller uses the setpoint of these loops as "manipulated variables".

In academia, there is widespread resentment against step response models - and in particular against their use in MPC controllers. Although there are valid arguments supporting this resentment, these are usually of little practical importance for asymptotically stable processes - although in some cases the computational burden can be reduced by using a state space model instead.

Step response *identification* is another matter. A step input has Laplace transform $u(s) = \frac{k}{s}$, and hence excites the process primarily at low frequencies. The resulting model can therefore be expected to be good only for the slow dynamics (low frequencies). If medium to high bandwidth control is desired for an MPC application, one should make sure that any identification experiment excites the process over the whole desired bandwidth range for the controller.

6.4 Updating the process model

The MPC controller essentially controls the *process model*, by optimizing the use of the inputs in order to remove the predicted deviation from some desired state (or output) trajectory. Naturally, good control of the *true process* will only be obtained if the process model is able to predict the future behaviour of the true process with reasonable accuracy. Model errors and unknown disturbances must always be expected, and therefore it will be necessary to update the process model to maintain good quality predictions of the future process behaviour.

The most general way of doing this is through the use of a state estimator, typically a Kalman filter. The Kalman filter may also be modified to estimate unmeasured disturbances or model parameters that may vary with time. The Kalman filter is described in advanced control engineering courses and in numerous textbooks, and will not be described further here.

The Kalman filter is, however, a tool that is valid primarily for *linear* problems, and may in some cases estimate state values that defy physical reason. For example, a Kalman filter may estimate a negative concentration of a chemical component in a process. In the rare cases where it is necessary to take physical constraints (and possibly non-linearities in the model) explicitly into account, it is possible to use an 'MPC-like', optimization-based approach to the estimation problem, resulting in what is known as 'moving horizon estimation'. To this author's knowledge, moving horizon estimation is not frequently used in industrial applications, and is to some extent still a research issue. A recent overview can be found in Allgöwer et al. [3].

For asymptotically stable systems, a particularly simple model updating strategy is possible for MPC formulations that only use process inputs and measurements in the formulation (i.e., when unmeasured states do not appear in the objective function or in the constraints). In such cases, it would be natural to calculate the predicted *deviations from the desired output trajectory* (which may be called, say, ψ_{dev}), rather than the predicted deviations from the desired *state* trajectory χ_{dev}. Then, the model can be 'updated' by simply adding the present difference between process output and model output to the model's prediction of the future outputs. This is known as a 'bias update', and is widespread in industrial applications. Note, however, that the bias update

- is only appliccable to asymptotically stable systems, and may result in poor control performance for systems with very slow dynamics, and that
- it may be sensitive to measurement noise. If a measurement is noisy, one should attempt to reduce the noise (typically by a simple low-pass filter) before calculating the measurement bias.

6.5 Feedforward from disturbances

With MPC it is very simple to include feedforward from measured disturbances, provided one has a model of how the disturbances affect the states/outputs. Feedforward is naturally used to counteract the future effects of disturbances on the controlled variables (it is too late to correct the present value). Thus, feedforward in MPC only requires that the effect on disturbances on the controlled variables are taken into account when predicting the future state trajectory in the absence of any control action. Thus, in the formulation developed above, feedforward from disturbances results from taking the disturbances into account when calculating χ_{dev}.

The benefit obtained by using feedforward will (as always) depend on what bandwidth limitations there are in the system for feedback control. Furthermore, effective feedforward requires both the disturbance and process model to be reasonably accurate.

6.6 Feasibility and constraint handling

For any type of controller to be acceptable, it must be very reliable. For MPC controllers, there is a special type of problem with regards to *feasibility* of the constraints. An optimization problem is *infeasible* if there exists no exists no set of values for the free variables in the optimization for which all constraints are fulfilled. Problems with infeasibility may occur when using MPC controllers, for instance if the operating point is close to a constraint, and a large disturbance occurs. In such cases, it need not be possible to fulfill the constraint at all times. During startup of MPC controllers, one may also be far from the desired operating point, and in violation of some constraints. Naturally, it is important that the MPC controller should not 'give up' and terminate when faced with an infeasible optimization problem. Rather, it is desirable that the performance degradation is predictable and gradual as the constraint violations increase, and that the MPC controller should effectively move the process into an operating region where all constraints are feasible.

If the constraints are *inconsistent*, i.e., if there exists no operating point where the MPC optimization problem is feasible, then the problem formulation in meaningless, and the problem formulation has to be modified. Physical understanding of the process is usually sufficient to ensure that the constraints are consistent. A simple example of an inconsistent set of constraints is if the value of the minimum value constraint for a variable is higher than the value of the maximum value constraint.

Usually, the constraints on the inputs (manipulated variables) result from true, physical constraints that cannot be violated. For example, a valve cannot be more than 100% open. On the other hand, constraints on the states/outputs often represent operational desireables rather than fundamental operational constraints. State/output constraints may therefore often be violated for short periods of time (although possibly at the cost of producing off-spec products or increasing the need for maintenance). It is therefore common to modify the MPC optimization problem in such a way that output constraints may be violated if necessary. There are (at least) three approaches to doing this modification:

1. Remove the state/output constraints for a time interval in the near future. This is simple, but may allow for unnecessarily large constraint violations. Furthermore, it need not be simple to determine for how long a time interval the state/output constraints need to be removed - this may depend on the operating point, the input constraints, and the assumed maximum magnitude of the disturbances.

2. To solve a separate optimization problem prior to the main optimization in the MPC calculations. This initial optimization minimizes some measure of how much the output/state constraints need to be moved in order to produce a feasible optimization problem. The initial optimization problem is usually a LP problem, which can be solved very efficiently.

3. Introducing *penalty functions* in the optimization problem. This involves modifying the constraints by introducing additional variables such that the constraints are always feasible for sufficiently large values for the additional variables. Such modified constraints are termed *soft constraint*s. At the same time, the objective function is modified, by introducing a term that penalizes the magnitude of the constraint violations. The additional variables introduced to ensure feasibility of the constraints then become additional free variables in the optimization. Thus, feasibility is ensured by increasing the size of the optimization problem.

The two latter approaches are both rigorous ways of handling the feasibility problem. Approach 3 has a lot of flexibility in the design of the penalty function. One may ensure that the constraints are violated according to a strict list of priorites, i.e., that a given constraint will only be violated when it is impossible to obtain feasibility by increasing the constraint violations for less important constraints. Alternatively, one may distribute the constraint violations among several constraints. Although several different penalty functions may be used, depending on how the magnitude of the constraint violations are measured, two properties are desireable:

- That the QP problem in the optimization problem can still be solved efficiently. This implies that the Hessian matrix for the modified problem should be positive definite, i.e., that there should be some cost on the *square* of the magnitude of the constraint violations.

- That the penalty functions are *exact*, which means that no constraint violations are allowed if the original problem is feasible. This is usually obtained by putting a sufficiently large weight on the magnitude of the constraint violations (i.e., the linear term) in the objective function.

The use of penalty functions is described in standard textbooks on optimization (e.g. [13]), and is discussed in the context of MPC in e.g. [11, 42, 27].

In addition to the problem with feasibility, hard output constraints may also destabilize an otherwise stable system controlled by an MPC controller, see [50]. Although this phenomenon probably is quite rare, it can easily be removed by using a soft constraint formulation for the output constraints [11]. The following section will discuss closed loop stability with MPC controllers in a more general context.

6.7 Closed loop stability with MPC controllers

The objective function in Eq. (31) closely resembles that of discrete-time Linear Quadratic (LQ) - optimal control. For stabilizable and detectable[17] systems, infinite horizon LQ-optimal control is known to result in a stable closed loop system. Note that the requirement for detectability does not only imply that unstable modes must be detectable from the physical measurements (i.e., that (C, A) is detectable), but also that the unstable modes must affect the objective function, i.e., $(Q^{1/2}, A)$ must be detectable.

With the stabilizability and detectability requirements fulfilled, a *finite horizon* LQ-optimal controller is stable provided the weight on the 'terminal state', S, is sufficiently large. How large S needs to be is not immediately obvious, but it is quite straight forward to calculate an S that is sufficiently large. In the MPC context, this can be done by designing a stabilizing state feedback controller K (typically, one would choose the infinite horizon LQ-optimal controller, obtained by solving a Riccati equation), and then

[17]Stabilizability is a weaker requirement than the traditional state controllability requirement, since a system is stabilizable if and only if all unstable modes are controllable, i.e., a system can be stabilizable even if some stable modes are uncontrollable. Similarly, a system is detectable if all unstable modes are observable.

calculate the S that gives the same contribution to the objective function that would be obtained by using the controller K, and summing the terms $(x_i - x_{ref,n})^T Q(x_i - x_{ref,n})$ from $i = n$ to infinity. Since the controller K results in an asymptotically stable system, this sum is finite, and hence S is finite. The value of S can be obtained by solving a discrete Lyapunov equation

$$S - (A + BK)^T S(A + BK) = Q$$

With a sufficiently large S, obtained as described above, the remaining requirement for obtaining closed loop stability is that constraints that are feasible over the horizon $n \leq i \leq n + j$ will remain feasible over an infinite horizon (assuming no new disturbances enter). Rawlings and Muske [36] have shown how to calculate a sufficiently large j. First arrange all state constraints for $n \leq i \leq n + j$ (including input constraints that effectively become state constraints through the assumption that a state feedback controller is used) on the form

$$\widetilde{H} x_i \leq \widetilde{h}$$

and diagonalize the autotransition matrix $A + BK$:

$$A + BK = T \Lambda T^{-1}$$

where Λ is a diagonal matrix (which may have complex-valued elements if $A + BK$ contains oscillatory modes). Then, a sufficiently large value of j can be calculated from

$$\begin{aligned} a &= \frac{\widetilde{h}_{\min}}{\overline{\sigma}(\widetilde{H})\gamma(T)\left\|x_n\right\|} \\ j &= \max\left\{0, \frac{\ln a}{\ln \lambda_{\max}}\right\} \end{aligned}$$

where $\widetilde{h}_{\min}$ is the smallest element in h, $\overline{\sigma}(\widetilde{H})$ is the maximum singular value of $\widetilde{H}$, $\gamma(T)$ is the condition number of T (ratio of largest to smallest singular value), $\|x_n\| = (x_n^T x_n)^{1/2}$, and is the magnitude of the largest element in Λ, i.e., the largest eigenvalue of $A + BK$.

The problem with the above estimate of j is that it depends on x_n, which can only be predicted when the result of the optimization is available. If the optimization is performed with too small a value for j, one will then have to re-calculate the optimization with an increased j, in order to be able to give any strict stability guarantee. In practice, a constant value for j based on simulations will be used.

The above results on how to find values for S and j to guarantee stability, are not very useful if, e.g., a step response model is used, since the values of the states are then unavailable. Step response-based MPC controllers therefore do not have a terminal state weight S, but rather extend the prediction of the outputs further into the future than the time horizon over which the inputs are optimized (corresponding to $n_p > n_u$ in the comments following Eq. (32). Although a sufficiently large prediction horizon n_p compared to the "input horizon" n_u will result in a stable closed loop system (the open loop system is assumed asymptotically stable, since a step response model is used), there is no known way of calculating the required n_p. Tuning of step-response based MPC controllers therefore typically rely heavily on simulation. Nevertheless, the industrial success of step response-based MPC controllers show that controller tuning is not a major obstacle in implementations.

6.8 Robustness of MPC controllers

The main advantage of MPC controllers lie in their ability to handle constraints. On the other hand, they may be sensitive to errors in the process model. There have been tales about processes which are controlled by MPC controllers when prices are high (and it is important to operate close to the process' maximum throughput), but are controlled by simple single-loop controller when prices are low (and production is lower, leading to no active constraints). The potential robustness problems are most easily understood for cases when no constraints are active, i.e., when we can study the objective function in Eq. (27) with H and c from Eq. (34). We then want to minimize

$$f(v) = v^T(\widehat{B}^T\widehat{Q}\widehat{B} + \widehat{P})v + \chi_{dev}^T\widehat{A}^T\widehat{Q}\widehat{B}v$$

with respect to v. The solution to this minimization can be found analytically, since no constraints are assumed to be active. We get[18]

$$v = -(\widehat{B}^T\widehat{Q}\widehat{B} + \widehat{P})^{-1}\widehat{B}^T\widehat{Q}\widehat{A}\chi_{dev}$$

[18]Note that $\widehat{Q} = \widehat{Q}^T$, and that the assumptions on Q, S and P ensures that $(\widehat{B}^T\widehat{Q}\widehat{B}+\widehat{P})$ is of full rank, and hence invertible.

Clearly, if the model contains errors, this will result in errors in $\widehat{B}$ and $\widehat{A}$, and hence the calculated trajectory of input moves, v, will be different from what is obtained with a perfect model. If the Hessian matrix $\widehat{B}^T\widehat{Q}\widehat{B}+\widehat{P}$ is *ill-conditioned*[19], the problem is particularly severe, since a small error in the Hessian can then result in a large error in its inverse. For a physical motivation for problems with ill-conditioning consider the following scenario:

- The controller detect an offset from the reference in a direction for which the process gain is low.
- To remove this offset, the controller calculates that a large process input is needed in the low gain input direction.
- Due to the model errors, this large input actually slightly "misses" the low gain input direction of the true process.
- The fraction of the input that misses the low gain direction, will instead excite some high gain direction of the process, causing a large change in the corresponding output direction.

Now, there are two ways of reducing the condition number of $\widehat{B}^T\widehat{Q}\widehat{B}+\widehat{P}$:

1. Scaling inputs and states in the process model, thereby changing $\widehat{B}$.
2. Modifying the tuning matrices $\widehat{Q}$ and $\widehat{P}$.

Scaling inputs and states (or outputs, if the objective function uses outputs instead of states) is essentially the same as changing the units in which we measure these variables. In some cases this sufficient, but some processes have inherent ill-conditioning that cannot be removed by scaling.

In theory, one may use non-zero values for all elements in the tuning matrices $\widehat{Q}$ and $\widehat{P}$, with the only restriction that $\widehat{Q}$ should be positive semi-definite[20] and $\widehat{P}$ should be positive definite (and hence both should be symmetric). However, little is known on how to make full use of this freedom in designing $\widehat{Q}$ and $\widehat{P}$, and in practice they are obtained from Q ,P and S as shown in Eq. (33), and typically Q and P are diagonal. It is common to try to reduce the ill-conditioning of the Hessian matrix by multiplying all elements of $\widehat{P}$ by the same factor. If this factor is sufficiently large,

[19] A matrix is ill-conditioned if the ratio of the largest singular value to the smallest singular value is large. This ratio is called the *condition number.*

[20] The lower right diagonal block of $\widehat{Q}$, corresponding to the terminal state weight S, should be strictly positive definite (and sufficiently large).

the condition number of the Hessian matrix will approach that of P - which can be chosen to have condition number 1 if desired. However, increasing all elements of $\widehat{P}$ means that the control will become slower in all output directions, also in directions which are not particularly sensitive to model uncertainty.

If the above ways of reducing the condition number of the Hessian matrix are insufficient or unacceptable, one may instead modify the process model such that the controller "does not see" offsets in the low gain directions. Inherent ill-conditioning (which cannot be removed by scaling) is typically caused by physical phenomena which make it difficult to change the outputs in the low gain direction. Fortunately, this means that disturbances will also often have a low gain in the same output direction. It may therefore be acceptable to ignore control offsets in the low gain output directions. In terms of the MPC formulation above, the controller can be forced to ignore the low gain directions by modifying $\widehat{B}$ by setting the small singular values of $\widehat{B}$ to zero. This is known as *singular value tresholding*, since we remove all singular values of $\widehat{B}$ that is smaller than some treshold. If we term this modified matrix $\widehat{B}$ for $\widehat{B}_m$, we find that the trajectory of input moves calculated by the (unconstrained) MPC optimization now becomes

$$v = -(\widehat{B}_m^T \widehat{Q} \widehat{B}_m + \widehat{P})^{-1} \widehat{B}_m^T \widehat{Q} \widehat{A} \chi_{dev} = -(\widehat{B}_m^T \widehat{Q} \widehat{B}_m + \widehat{P})^{-1} \chi_m$$

Note that the conditioning of the Hessian matrix is not improved by setting the small singular values of $\widehat{B}$ to zero, but the vector χ_m does not show any control offset in the corresponding output directions, and hence the vector v will contain no input moves in the corresponding input directions.

Singular value tresholding is effective in improving robustness to model errors, but it clearly causes nominal control performance (the performance one would get if the model is perfect) to deteriorate, since the controller ignores control offsets in some output directions. Removing too many singular values from $\widehat{B}$ will result in unacceptable control performance.

6.9 Using rigorous process models in MPC

Most chemical processes are inherently nonlinear. In some cases, rigorous dynamical models based on physical and chemical relationships are available, and the process engineers may wish to use such a model in an MPC controller. This would for instance have the advantage of automatically updating the model when the process is moved from one operating point to another.

However, to optimize directly on the rigorous model is not straight forward. The non-linearity of the model typically results in optimization problems that are non-convex. Optimization of non-convex problems is typically *a lot* more time consuming than optimization of convex problems and the time required to find a solution can vary dramatically with changing operating point or initial states. This means that direct optimization of non-linear models is usually ill-suited for online applications like MPC. Furthermore, it is often the case that the most important 'non-linearities' in the true system are the constraints, which are handled effectively by MPC.

This does not mean that rigorous models cannot be utilized by MPC controllers, but it means that one can make only partial use of such models. The idea is to utilize these models to the extent that the available time permits. One may then approximate the true optimization problem by a modified, convex problem, or a series of such problems.

Predict using the rigorous model. The simplest way of (partially) accounting for non-linearity in the process model, is to calculate the deviation from the desired state (or output) trajectory from a rigorous, non-linear model, whereas the other parts of the optimization formulation uses a linearized model. In this way, the calculated input trajectory v will to some extent account for the non-linearities.

Line search If greater accuracy is needed, one may do a line search

using the non-linear model to optimize what multiple of v should be implemented, i.e., perform a search to optimize (while taking the constraints into account)

$$\min_{\alpha} f(x,u) = \min_{\alpha} f(x_0, u_{ref} + \alpha v) \tag{43}$$

where α is a positive real scalar. Such line searches are a standard part of most non-linear optimization methods, and are covered in many textbooks on optimization e.g. in [13]. When performing the minimization in Eq. (43) above, the full non-linear model is used to calculate future states from $(x_0, u_{ref} + \alpha v)$.

Iterative optimization. Even with the optimal value of α, one probably has not found the optimal solution to the original non-linear optimization problem. Still better solutions may be found by an iterative procedure, where the predicted deviation from the desired state trajectory x_{ref} is found using the best available estimate of the future input trajectory. That is, for iteration number k, use the model to calculate the resulting vector

$\chi_{dev,k}$ when the input trajectory $u_{ref} + v_t$ is applied, where $v_t = \sum_{l=0}^{k-1} v_l$, and minimize

$$\min_{v_k} f(v) = (v_t + v_k)^T (\widehat{B}^T \widehat{Q} \widehat{B} + \widehat{P})(v_t + v_k) + \chi_{dev,k}^T \widehat{A}^T \widehat{Q} \widehat{B}(v_t + v_k)$$

subject to constraints that should be modified similarly. It is also assumed that a line search is performed between each iteration. The iterations are initialized by setting $v_0 = 0$, and are performed until the optimization converges, or until the available time for calculations is used up. The iterative procedure outlined above need not converge to a globally optimal solution for the original problem, it may end up in a local minimum. Furthermore, there is no guarantee that this is a particularily efficient way of solving the original optimization problem (in terms of the non-linear model). It does, however, have the advantage of quickly finding reasonable, and hopefully feasible, input sequences. Even if the optimization has to terminate before the optimization has converged, a 'good' input has been calculated and is available for implementation on the process.

Linearize around a trajectory. If the operating conditions change significantly over the time horizon (n) in the MPC controller, the linearized model may be a reasonable approximation to the true process behaviour for only a part of the time horizon. This problem is relatively rare when constant reference values are used, but may be relevant when moving from one operating point to another. It is then possible to linearize the process around the predicted process trajectory ($x_{ref} + \chi_{dev}$) rather than around a constant state. One then gets a time-varying (but still linear) model, i.e., a "new model" for each time interval into the future. Conceptually, linearizing around a trajectory does not add much complexity compared to linearizing around a constant state, but it does add significantly to the notational complexity that is necessary in the mathematical formulation of the optimization problem. Furthermore, analytical representations of the linearized models are typically not available, and the linearization has to be performed by numerically perturbing the process around the predicted process trajectory. This can clearly add significantly to the computational burden. Linearizing around a trajectory can be combined with iterative optimization as outlined above - which would further add to the computational burden.

7 Controller Performance Monitoring and Diagnosis

7.1 Introduction

It is a sad fact that many control loops in industrial processes actually degrade system performance, by increasing the variability in the controlled variable rather than decreasing it. Still more control loops do actually work, but are very far from optimal. Some causes for poor controller performance are:

- Operating conditions have changed after the controller was tuned.
- The actual process has changed, some process modifications have been made after the controller was tuned.
- The controller has never actually been tuned, it is still using the manufacturer's default tuning parameters.
- A poor (or even inconsistent) control structure, causing severe interactions between control loops.
- Some equipment in the control loop may be in need of maintenance or replacement, e.g., faulty measurements, control valves with excessive stiction, severe fouling in heat exchangers, etc.

There are many reasons why such a situation may be allowed to last. Often, plant operators are aware of what parts of the process are oscillating or show large control offsets. However, this information often stays with the operators, and they learn to cope with the process as it is. The typical operator will lack the competence to assess whether the observed control performance is much worse than what should be expected. When asked a general question about whether control of the process is acceptable, they may therefore very well confirm that the control is good even if that is not the case.

The automation department of a large plant is normally very small. The typical automation department is fully occupied with keeping the various automation and control system in operation, with little time for improving the control system. Most industrial automation engineers are therefore also trained to keep the control system running, and have little relevant background for evaluating controller performance or improving controllers. After an initial commissioning phase, most controllers are therefore "left alone" for long periods.

Considering the large number of control loops in an industrial plant, there is a need for tools which ensure efficient use of what little time is available for improving the control system, that is, tools which help the engineer to

- focus on where the control problems are most acute
- quickly assess whether significant improvements are easily achievable, e.g. by retuning the controller
- diagnose the cause for poor control performance.

The sections below will present some of the tools and results that are available for helping the engineer with the above tasks. We will first consider the detection and diagnosis of oscillating control loops, and thereafter discuss the assessment of controller performance.

7.2 Detection of oscillating control loops

For the trained human eye, detection of oscillations may seem a trivial task. However, it is far from trivial to define and describe oscillations in a typical signal from a process plant in such a way that it can reliably be automated (in either on-line or off-line tools). We will here present a few tools that have been proposed, but first present some statistical tools. It is assumed that the signals under study are stable, as otherwise the control loops in question will have to be taken out of service (and it should then be apparent that the control loop needs attention).

7.2.1 The autocorrelation function

The autocorrelation function is essentially a measure of how closely the values of a variable, when measured at different times, are correlated. For a variable y and a data set of N datapoints, the autocorrelation function is given by

$$\rho_k = \frac{\sum_{t=1}^{N-k}(y_t - \overline{y})(y_{t+k} - \overline{y})}{\sum_{t=1}^{N}(y_t - \overline{y})^2}$$

The autocorrelation function is 1 for lag 0, that is, $\rho_0 = 1$. For stable signals, it generally decays with increasing lags, whereas it will oscillate for systematically oscillating signals, and a periodic signal will have a periodic autocorrelation function.

In principle, one should be able to detect oscillations directly from the autocorrelation function. However, it need not be so straight forward if the signal contains multiple frequencies, measurement noise, assymmetric oscillations, etc. Nonlinear effects may also introduce oscillations at frequencies that are multiples of the base oscillation frequency. Nevertheless, one of the methods presented below will make direct use of the autocorrelation function.

7.2.2 The power spectrum

The power spectrum results from a Fourier transform of the autocorrelation function, and in essence it is the frequency domain equivalent of the autocorrelation function. If the signal oscillates at a particular frequency, the power spectrum will have a peak at that frequency. An oscillation that does not decay with time, will have a very large peak at that frequency in the power spectrum. The problems of using the power spectrum for oscillation detection are similar to those of using the autocorrelation function. Instead of the power spectrum having a single spike at the oscillating frequency, the signal may be so corrupted by noise and nonlinear effects that it looks more like the back of a hedgehog.

7.2.3 The method of Miao and Seborg

Miao and Seborg[32] uses the autocorrelation function to detect oscillations. It calculates a somewhat non-standard 'decay ratio', as illustrated in Fig. 17.

The Miao-Seborg oscillation index is simply the ratio given by $R = a/b$. Miao and Seborg propose a treshold value of $R = 0.5$, a larger value will indicate (unacceptable) oscillations. Little justification is provided for this measure. In particular, it is not explained why this measure is better than simply comparing the magnitude of neighbouring peaks in the autocorrelation function.

Nevertheless, industrial experience appears to be favourable, and oscillations are detected with reasonable reliability. Some drawbacks are

- it is somewhat complicated for on-line oscillation detection, it is better suited for offline analysis of batches of data.

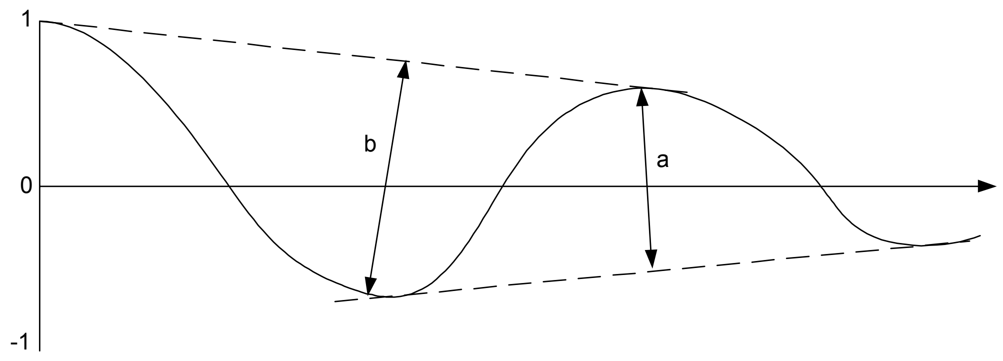

Figure 17: Calculation of the Miao-Seborg oscillation index from the autocorrelation function.

- it does not take the amplitude of oscillations into account. Some oscillations of small amplitude may be acceptable.
- it assumes that the oscillations are the main cause of variability in the measured variable. If a control loop experiences frequent (and irregular) setpoint changes of magnitude larger than the amplitude of the oscillations, it may fail to detect the oscillations.

7.2.4 The method of Hägglund

Hägglunds measure[18] may be said to be a measure for control performance rather than an oscillation detection method. The basic idea behind the measure is that the controlled variable in a well-functioning control loop should fluctuate around the setpoint, and that long periods on one side of the setpoint is a sign of poor tuning.

Hägglund's performance monitor looks at the control error $e(t) = r(t) - y(t)$, and integrates the absolute value of $e(t)$ for the period between each time this signal crosses zero:

$$IAE = \int_{t_{i-1}}^{t_i} |e(t)|\, dt$$

where t_{i-1} and t_i are the times of two consequtive zero crossings. Whenever this measure increases beyond a treshold value, a counter is incremented, and an alarm is raised when the counter passes some critical value. It is shown in [18] how a forgetting factor can be used to avoid alarms from well-functioning loops which are exposed to infrequent, large disturbances (or setpoint changes).

Critical tuning parameters for this monitoring method are the IAE treshold value and the counter alarm limit. Typical choices for the IAE treshold value are

$$\begin{aligned} IAE_{\text{lim}} &= 2a/\omega_u \\ IAE_{\text{lim}} &= aT_I/\pi \end{aligned}$$

where a is an acceptable oscillation magnitude, ω_u is the ultimate frequency (the oscillation frequency found in a closel loop Ziegler Nichols experiment), and T_I is the integral time in a PI(D) controller. The more rigorous of the two treshold values is the first, and ω_u would be available if the loop was tuned with e.g. Hägglund's relay-based autotuning procedure. However, often ω_u will not be available, and the second expression for IAE_{lim} will then have to be used - this expression is intended to work as a reasonable approximation of the first expression for IAE_{lim} for a reasonably tuned loop. Naturally, this may be misleading if the cause of poor control performance is poor choice of controller tuning parameters.

The counter alarm limit is simply a tradeoff between the sensitivity of the monitoring method and the rate of "unnecessary" alarms. This monitoring method is

- Simple and appliccable for on-line implementation.
- It takes oscillation amplitude into account - it is only affected by small oscillations if the oscillation period is very long.
- Some tuning of the monitoring method must be expected. The guidelines for choosing IAE_{lim} may be misleading in some cases.

7.2.5 The method of Forsman and Stattin

This method also looks at the controll offset $e(t) = r(t) - y(t)$, but it is strictly and oscillation detection method and not a general performance monitor. Forsman and Stattin [14] proposes comparing both the areas between

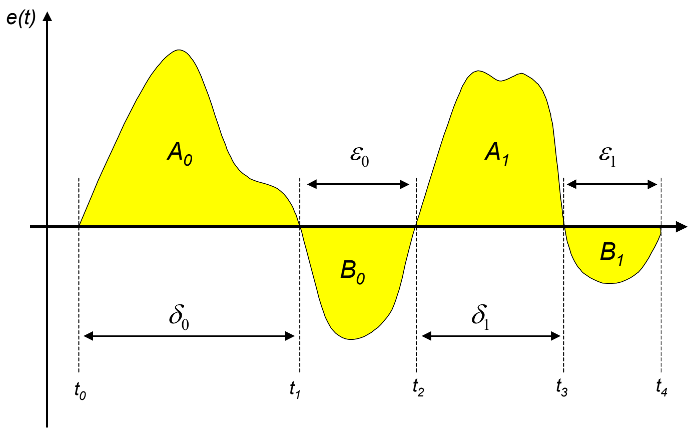

Figure 18: The oscillation detection method of Forsman and Stattin [14].

the control offset and zero and the time span that the offset has the same sign. However, the resulting area and time span is not compared with the immediately previous area/timespan (when the control error had opposite sign), rather the comparison is made with the preceding period when the control offset had the same sign. This is illustrated in Fig. 18.

The method uses two tuning constants α and γ, that both should be in the range between 0 and 1, and simply counts the number of times h_A in a data set that

$$\alpha < \frac{A_{i+1}}{A_i} < \frac{1}{\alpha} \text{ and/or } \gamma < \frac{\delta_{i+1}}{\delta_i}$$

and the number of times h_B that

$$\alpha < \frac{B_{i+1}}{B_i} < \frac{1}{\alpha} \text{ and/or } \gamma < \frac{\varepsilon_{i+1}}{\varepsilon_i}$$

The oscillation index is then given by $h = (h_A + h_B)/N$, where N is the number of times in the data set that the control offset crosses zero.

Forsman and Stattin recommend closer examination of loops having $h > 0.4$, and if $h > 0.8$ a very clear oscillative pattern can be expected.

7.2.6 Pre-filtering data

All three methods presented above may be ineffective for noisy data, and both Miao and Seborg [32] and Forsman and Stattin [14] discuss pre-filtering the data with a low pass filter to reduce the noise. Clearly, the filter should be designed to give a reasonable tradeoff between noise and oscillation detection in the frequency range of interest. The interested reader should consult the original references for a more comprehensive treatment of this issue.

7.3 Oscillation diagnosis

Once an oscillating control loop has been detected, it is naturally of interest to find the cause of the oscillations, in order to come up with some effective remedy. There is no general solution to the diagnosis problem, the proposed methods can at best handle parts of the problem. We will present a manual diagnosis procedure proposed by Hägglund [18], and a passive procedure (that may be automated) for detecting valve stiction proposed by Horch [23].

7.3.1 Manual oscillation diagnosis

In [18], Hägglund proposes the manual oscillation diagnosis procedure presented in Fig. 19

The main problem with this procedure is the assumption that if the oscillation (in the controlled variable) stops when the controller in a particular loop is put in manual, then the oscillation is caused by that loop. Often, oscillations arise from multivariable interactions between loops, and the oscillation will then stop when any one of these loops are put in manual. Typically, the loop which receives the "blame" for the oscillations will be detuned (made slower). Therefore, the results of this procedure willdepend on the order in which the loops are examined. If several loops show a similar oscillation pattern, one should therefore first examine the loop for which slow control is more acceptable.

The procedure is also a little short on examining other instrumentation problems than valve friction, e.g., valve hysteresis, measurement problems, etc. Furthermore, the procedure gives no proposals for how to eliminate external disturbances. Clearly, the solution will be very dependent on the

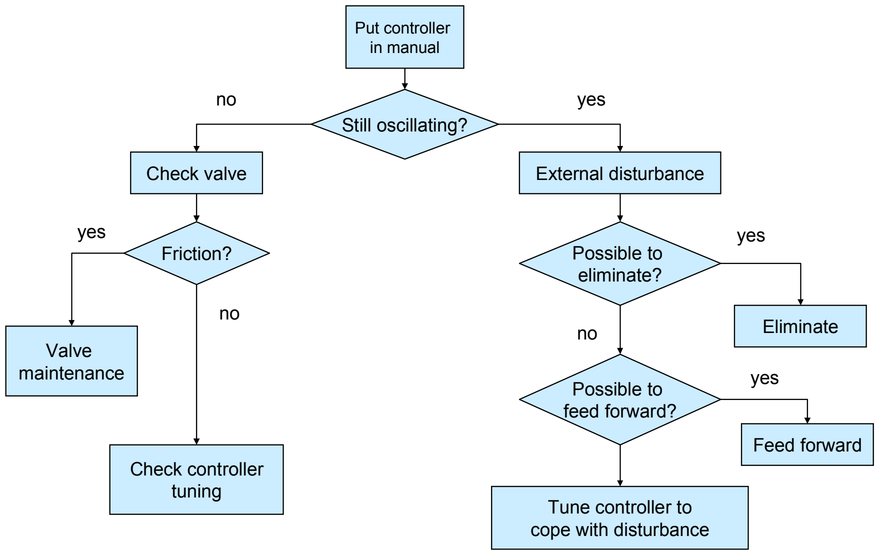

Figure 19: Hägglund's manual oscillation diagnosis procedure.

particular process, but typically it will involve modifying the process or the control in other parts of the process.

7.3.2 Detecting valve stiction

A commonly occuring problem with valves is that they can have a tendency to stick due to stiction (short for 'static friction'). Once the controller applies sufficient force to overcome the stiction and move the valve, the friction force drops dramatically (since the 'dynamic' friction is much smaller than the static friction). This results in a large net force acting on the valve stem, causing a sudden move of it. It is well known that such stiction can cause oscillations.

Horch [23] have developed a method for detecting stiction, based on measurements of the controlled variable and the controller output. The method assumes that the controller has integral action. The integral action will steadily increase the controller output, until the valve suddenly "jumps" to a new position. Persisten oscillations often result when the valve jumps to far, so that the controller has to stop the valve movement and move it in the opposite direction. Stopping the valve causes it to stick again, causing the sequence of events to repeat.

When there is problems with valve stiction, the controller output signal typically has a sawtooth shape. The controlled variable is typically almost like a square wave, especially if the dominant time constant of the process (in open loop) is much shorter than the period of oscillation.

Horch found that the cross-correlation function between controller output and controlled variable typically is an odd function[21] for a system oscillating due to stiction. On the other hand, if the oscillation is due to external disturbances, the cross-correlation function is normally close to an even function. Unstable loops oscillating with constant amplitude (due to input saturation) also have an even cross-correlation function.

For a data set with N data points, the cross-correlation function between u and y for lag τ (where τ is an integer) is given by

$$r_{uy}(\tau) = \frac{\sum_{k=k_0}^{k_1} u(k)y(k+\tau)}{\sum_{k=1}^{N} u(k)y(k)} \tag{44}$$

where

$$\begin{aligned} k_0 &= 1 \text{ for } \tau \geq 0 \\ k_0 &= \tau + 1 \text{ for } \tau < 0 \\ k_1 &= N - \tau \text{ for } \tau \geq 0 \\ k_1 &= N \text{ for } \tau < 0 \end{aligned}$$

Note that the denominator in Eq. (44) is merely a normalization, giving $r_{uy}(0) = 1$, it is not necessary for the stiction detection method.

Horch' stiction detection method has been found to work well in most cases. However, it fails to detect stiction in cases where the dominant time constant of the (open loop) process is large compared to the observed period of oscillation. In such cases the cross-correlation function will be approximately even also for cases with stiction. This problem is most common with integrating processes (e.g., level control loops), but may also occur for other processes with slow dynamics.

Horch [24] has recently proposed an alternative method for stiction detection for integrating processes. Industrial experience with this alternative method is not known. However, it is patented by ABB.

7.4 Control loop performance monitoring

7.4.1 The Harris Index

[21] Reflecting the 90° phase shift due to the interal action in the controller.

The most popular index for monitoring controller performance has been named after T. J. Harris. Control loop performance monitoring has received much attention since his publication of an influential paper on the subject [19], although similar ideas have been proposed earlier, by e.g., Fjeld [12]. The Harris' index simply compares the observed variance in the controlled variable with the best achievable variance. The observed variance is easily calculated from on-line data. The beauty of the method lies in that only modestly restrictive assumptions about the are necessary in order to estimate the minimum achievable variance from available on-line data.

The necessary assumptions are:

1. The deadtime from manipulated variable u to controlled variable y must be known or estimated.

2. The process is asymptotically stable.

3. The process does not have any inverse response[22].

Assumptions 2 and 3 above may be relaxed, if a sufficiently accurate process model is available, see Tyler and Morari [48].

When assumptions 1-3 are fulfilled, a minimum variance controller may be used, and as the name says, this controller would achieve the minimum variance in the output. The minimum variance controller will not be derived here, but it is described in many textbooks on stochastic control theory. All we need is the following observations:

- No control action can influence the controlled variable before at least one deadtime has passed.

- The minimum variance controller will remove all autocorrelation in the controlled variable for time lags greater than the deadtime.

Thus, if we have an impulse response model for the effect of the (unknown) disturbance on the controlled variable with the existing controller

$$y_k = \sum_{i \geq 0} h_i d_{k-i}$$

[22]In terms of systems theory, the (discrete time) process should not have any zeros on or outside the unit disk. This corresponds to zeros in the right half plane for continuous-time systems.

we know that h_i is unaffected by feedback for $i < \delta$, where δ is the deadtime (in number of sample intervals), whereas the minimum variance controller would achieve $h_i = 0$ for $i \geq \delta$. Thus, the minimum achievable variance in y is

$$\sigma_{y,mv}^2 = (1 + h_1^2 + h_2^2 + \cdots + h_{\delta-1}^2)\sigma_d^2 \tag{45}$$

where we have selected $h_0 = 1$, since this is equivalent to scaling the disturbance variance σ_d^2.

7.4.2 Obtaining the impulse response model

In order to identify a model for the effect of the unknown disturbance on the controlled variable, we must first select a model structure. We will use an autoregressive (AR) model, where we assume that the disturbance d is a zero mean white noise:

$$y_k + a_1 y_{k-1} + a_2 y_{k-2} + \cdots = d_k$$

or, in terms of the *backwards shift operator* z^{-1}:

$$(1 + a_1 z^{-1} + a_2 z^{-2} + a_3 z^{-3} + \cdots)y_k = A(z^{-1})y_k = d_k$$

Now, the AR model is very simple, and one may therefore need a high order for the polynomial $A(z^{-1})$in order to obtain a reasonably good model. One therefore runs the risk of "fitting the noise" instead of modelling system dynamics. It is therefore necessary to use a data set that is much longer than the order of the polynomial $A(z^{-1})$. However, if a sufficiently large data set is used (in which there is significant variations in the controlled variable y), industrial experience indicate that acceptable models for the purpose of control loop performance monitoring is often obtained when the order of the polynomial $A(z^{-1})$ is 15-20. The AR model has the advantage that a simple least squares calculation is all that is required for finding the model, and this calculation may even be performed recursively, i.e., it is appliccable for on-line implementation.

We will here only consider off-line model identification. The expected value of the disturbance d is zero, and thus we have for a polynomial $A(z^{-1})$

of order p and a data set of length N with index k denoting the most recent sample

$$\begin{bmatrix} y_{k-1} & y_{k-2} & \cdots & y_{k-p+1} & y_{k-p} \\ y_{k-2} & y_{k-3} & \cdots & y_{k-p} & y_{k-p-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ y_{k-N+p} & y_{k-N-1+p} & \cdots & y_{k-N+2} & y_{k-N+1} \\ y_{k-N-1+p} & y_{k-N-2+p} & \cdots & y_{k-N+1} & y_{k-N} \end{bmatrix} \begin{bmatrix} a_1 \\ \vdots \\ a_p \end{bmatrix}$$

$$= \quad - \begin{bmatrix} y_k \\ y_{k-1} \\ \vdots \\ y_{k-N+p+1} \\ y_{k-N+p} \end{bmatrix} + \begin{bmatrix} d_k \\ d_{k-1} \\ \vdots \\ d_{k-N+p+1} \\ d_{k-N+p} \end{bmatrix}$$

$$\Updownarrow$$

$$Y\underline{a} = -\underline{y} + \underline{d}$$

where the underbars are used to distinguish vector-valued variables from scalar elements. The expected value of the disturbance d is zero, and thus the model is found from a least squares solution after setting $\underline{d}= 0$:

$$\underline{a} = -(Y^T Y)^{-1} Y^T \underline{y}$$

After finding $\underline{a}$, an estimate of the noise sequence is simply found from $\underline{d}= Y\underline{a}+\underline{y}$, from which an estimate of the disturbance variance σ_d^2 can be found. Having found the polynomial $A(z^{-1})$, the impulse response coefficients h_i are found from

$$y_k = \frac{1}{A(z^{-1})} d_k = H(z^{-1}) d_k$$

using polynomial long division. Here $H(z^{-1}) = 1 + h_1 z^{-1} + h_2 z^{-2} + h_3 z^{-3} + \cdots$.

7.4.3 Calculating the Harris index

The Harris index is the ratio of the observed variance to the minimum achievable variance. The minimum achievable variance can be calculated from

Eq. (45) above, using the identified impukse response coefficients and the estimated disturbance variance

$$\sigma_d^2 = \frac{1}{N-1}\sum_{i=1}^{N}\left(d_i - \overline{d}\right)^2$$

where $\overline{d}$ is the mean value of the disturbance, which is zero by assumption.

The observed variance of the controlled variable can be computed similarly. However, if there is a persistent offset in the control loop, i.e., if the mean value of the controlled variable deviates from the reference, this should also be reflected in a measure of control quality. Hence, a modified variance should be used which accounts for this persistent offset

$$\sigma_{y,o}^2 = \frac{1}{N-1}\sum_{i=1}^{N}(y_i - y_{ref})^2$$

If there is a persistent offset from the reference, the modified variance $\sigma_{y,m}^2$ will always be larger than the true variance σ_y^2, and the Harris index becomes

$$H_I = \frac{\sigma_{y,o}^2}{\sigma_{y,mv}^2}$$

7.4.4 Modification to Harris' index

Despite the theoretical elegance of the derivation of the minimum variance controller, the minimum variance controller is generally not a realistic choice for a controller in a real application. This is because it is sensitive to model errors, and may use excessive moves in the manipulated variable. It *does* provide an absolute lower bound on the theoretically achievable variance, but it is nevertheless of interest to have a control quality measure which compares the actual performance to something (hopefully) more realistic.

One such modification is to assume that the 'ideal' controller does not immediately remove the effect of disturbances after one deadtime has passed, but rather that the effect of the disturbance decays as a first order function after the deadtime has passed. If we assume that this decay is described by the parameter μ ($0<\mu<1$), so that the ideal response to disturbances against which performance is measured would be

$$y_{k,\text{mod}} = \sum_{i=0}^{\delta-1} h_i d_{k-i} + \sum_{i=\delta}^{\infty} h_{\delta-1}\mu^{i-\delta+1} d_{k-i}$$

which results in a modified 'benchmark variance'

$$\sigma^2_{y,\text{mod}} = \sigma^2_{y,mv} + \frac{\mu^2}{1-\mu^2}\sigma^2_d$$

The modified contorl performance index then simply becomes

$$H_{I,\text{mod}} = \frac{\sigma^2_{y,o}}{\sigma^2_{y,\text{mod}}}$$

This modified Harris index is proposed by Horch and Isaksson [25] and Kozub [30]. Horch and Isaksson also provide some guidelines for how to specify the tuning factor μ. They find that if one wishes to account for a possible error in the estimated deadtime of ± 1 sample interval, and still require a gain margin of 2 for the 'ideal closed loop', this corresponds to choosing $\mu < 0.5$. It is also recommended to have a realistic attitude to how much the dynamics of the closed loop system can be speeded up, compared to the dynamics of the open loop process. Horch and Isaksson argue that it is unrealistic to speed up the system by a factor of more than 2-4. If we denote the open loop dominant time constant τ_{ol}, and the desired closed loop time constant is τ_{ol}/v, then the parameter μ should be chosen as

$$\mu = \exp\left(-\frac{vT_s}{\tau_{ol}}\right)$$

where T_s is the sampling interval for the control system.

7.4.5 Comments on the use of the Harris index

Before screening for poorly performing loops using the Harris index (or preferably the modified version presented above), one should first remove any persistently oscillationg loops, as these will certainly require attention.

It is important to realize that the Harris index is a *relative* measure of control quality. Thus, if a process is modified to improve controllability, e.g., by installing a new measurement with less deadtime, the Harris index may well get worse even if the actual performance improves significantly. This is of course because the observed variances before and after the process modifications are not compared against the same minimum variance.

The Harris index is appliccable to systems where the deadtime is the main factor limiting bandwidth and control performance. It was mentioned earlier that there are available modifications which allow consistent assessment of loops controlling an unstable process, or processes with inverse response (zero outside the unit disc). However, these modifications require much more detailed process knowledge than the basic Harris index. Similarly, the Harris index is not appliccable to control loops where the manipulated variable is in saturation much of the time.

Despite these limitations, the Harris index is appliccable to many control loops in most chemical processes.

7.4.6 Open issues in performance monitoring.

There are many open issues in performance monitoring and diagnosis. This will probably always be the case, since one attempts to evaluate the system behaviour without upsetting the system. Many aspects of the system behaviour will therefore necessarily remain unknown. Among the issues in need of further study are

- Performance monitoring in multivariable systems. Most systems are inherently multivariable, even if is is controlled by a set of monovariable control loops. Although the concept of the minimum variance controller can be extended to multivariable systems, there are problems regarding how to determine the 'interactor matrix', i.e., the multivariable deadtime matrix. Furthermore, in multivariable systems there will always be tradeoffs between variance in different controlled variables, and the relevance of any multivariable performance measure will depend critically on such tradeoffs.

- Performance monitoring for constrained systems. Model predictive control is particularly adept at handling constrained systems. The optimization criterion in the MPC controller will typically be designed to reflect the designers concept of optimal performance. Poor performance will then typically stem from model inaccuracies (including equipment in need of repair), changing noise levels, etc., and any performance monitor should probably attempt to pinpoint such problems.

- Diagnosis of oscillation in multi-loop systems. Hägglunds method presented above was critisized for not distinguishing properly between oscillations due to disturbances that enter from outside of the system, and oscillations due to interactions between multiple loops. Further work is needed on this topic.

- Diagnosis of malfunctioning valves. The method of Horch is in many cases able to detect valve problems due to stiction. However, other types of non-ideal valve behaviour are not covered, e.g., valve hysteresis.

References

[1] K. Aastrom, C. C. Hang, P. Persson, and W. K. Ho. Towards intelligen PID control. *Automatica*, 28(1):1–9, 1992.

[2] K. J. Aastrom and T. Hägglund. *Automatic Tuning of PID Controllers.* ISA, Research Triangle Park, NC, USA, 1988.

[3] F. Allgöwer, T. A. Badgewell, J. S. Qin, J. B. Rawlings, and S. J. Wright. Nonlinear model predictive control and moving horizon estimation - an introductory overview. In *Advances in Control. Highlights of the ECC'99.* Springer, 1999.

[4] K. Åström and T. Hägglund. Automatic tuning of simple regulators with specifications on phase and amplitude margins. *Automatica*, 20(5):645–651, 1984.

[5] J. G. Balchen. The stability of 2x2 multivariable control systems. *Modeling, Identification and Control*, 11:97–108, 1990.

[6] J. G. Balchen and K. I. Mummé. *Process Control. Structures and Applications.* Blackie Academic & Professional, Glasgow, Scotland, 1988.

[7] H. W. Bode. *Network Analysis and Feedback Amplifier Design.* Van Nostrand, Princeton, New Jersey, USA., 1945.

[8] R. D. Braatz and M. Morari. Minimizing the euclidian condition number. *SIAM Journal on Control and Optimization*, 32(6):1763–1768, 1994.

[9] E. H. Bristol. On a new measure of interactions for multivariable process control. *IEEE Transactions on Automatic Control*, AC-11:133–134, 1966.

[10] G. Cohen and G. Coon. Theoretical considerations of retarded control. *Trans. ASME*, 75:827–834, 1953.

[11] N. M. C. de Olivieira and L. T. Biegler. Constraint handling and stability properties of model-predictive control. *AIChE Journal*, 40(7):1138–1155, July 1994.

[12] M. Fjeld. On-line modelling of the performance of control systems. Annual Pittsburgh conference on modeling and simulation, 1981.

[13] R. Fletcher. *Practical Methods of Optimization.* John Wiley & Sons, 2nd edition, 1987.

[14] K. Forsman and A. Stattin. A new criterion for detecting oscillations in control loops. In *Proceedings of the European Control Conference*, 1999.

[15] J. S. Freudenberg and D. P. Looze. *Frequency Domain Properties of Scalar and Multivariable Feedback Systems.* Springer-Verlag, Berlin, Germany, 1988.

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

[17] P. Grosdidier, M. Morari, and B. R. Holt. Closed-loop properties from steady-state gain information. *Industrial and Engineering Chemistry Process Design and Development*, 24:221–235, 1985.

[18] T. Hägglund. A control-loop performance monitor. *Control Eng. Practice*, 3(11):1543–1551, 1995.

[19] T. J. Harris. Assessment of control loop performance. *Can. J. Chem. Eng.*, 67:856–861, October 1989.

[20] K. Havre. *Studies on Controllability Analysis and Control Structure Design.* PhD thesis, Norwegian University of Science and Technology, 1998.

[21] B. R. Holt and M. Morari. Design of resilient processing plants v - the effect of deadtime on dynamic resilience. *Chemical Engineering Science*, 40:1229–1237, 1985.

[22] B. R. Holt and M. Morari. Design of resilient processing plants VI - the effect of right half plane zeros on dynamic resilience. *Chemical Engineering Science*, 40:59–74, 1985.

[23] A. Horch. A simple method for oscillation diagnosis in process control loops. *Control Engineering Practice*, pages 1221–1231, 1999.

[24] A. Horch. *Condition Monitoring of Control Loops*. PhD thesis, Royal Institute of Technology, Stockholm, Sweden, 2000.

[25] A. Horch and A. J. Isaksson. A modified index for control performance assessment. In *Proceedings of the American Control Conference*, pages 3430–3434, 1998.

[26] M. Hovd. *Studies on Control Structure Selection and Design of Robust Decentralized and SVD Controllers*. PhD thesis, Department of Chemical Engineering, University of Trondheim-NTH, 1992.

[27] M. Hovd and R. D. Braatz. Handling state and output constraints in MPC using time-dependent weights. In *Proceedings of the American Control Conference*, 2001.

[28] M. Hovd, J. H. Lee, and M. Morari. Truncated step response models for model predictive control. *J. Proc. Cont.*, 3(2):67–73, 1993.

[29] M. Hovd and S. Skogestad. Improved independent design of robust decentralized controllers. *J. Proc. Cont.*, 3(1):43–51, 1993.

[30] D. J. Kozub. Controller performance monitoring and diagnosis: Experiences and challenges. In *Chemical Process Control*, pages 83–96, Tahoe City, California, 1996.

[31] H. M. and R. D. Braatz. On the computation of disturbance rejection measures. In *Preprints ADCHEM*, pages 63–68, Pisa, Italy, June 2000.

[32] T. Miao and D. E. Seborg. Automatic detection of excessively oscillating feedback control loops. In *IEEE Conference on Control Applications - Proceedings, Vol. 1.*, pages 359–364, 1999.

[33] M. Morari and E. Zafiriou. *Robust Process Control*. Prentice-Hall Inc., Englewood Cliffs, New Jersey, USA., 1989.

[34] C. N. Nett and V. Manousiouthakis. Euclidian condition and block relative gain - connections, conjectures and clarifications. *IEEE Transactions on Automatic Control*, AC-32(5):405–407, 1987.

[35] A. Niederlinski. A heuristic approach to the design of linear multivariable interacting control systems. *Automatica*, 7:691–701, 1971.

[36] J. B. Rawlings and K. R. Muske. The stability of constrained receding horizon control. *IEEE Transactions on Automatic Control*, 38(10):1512–1516, 1993.

[37] J. E. Rijnsdorp. Interaction in two-variable control systems for distillation columns-i. *Automatica*, 1:15–28, 1965.

[38] D. E. Rivera, M. Morari, and S. Skogestad. Internal model control. 4. PID controller design. *Ind. Eng. Chem. Process Des. Dev.*, 25:252–265, 1986.

[39] H. H. Rosenbrock. *State Space and Multivariable Theory.* Nelson, London, 1970.

[40] N. Sandell, P. Varaiya, M. Athans, and M. G. Safonov. Survey of decentralized control methods for large scale systems. *IEEE Trans. on Automatic Control*, AC-32(2):108–128, 1978.

[41] T. Schei. A method for closed-loop automatic tuning of PID controllers. *Automatica*, 28(3):587–591, 1992.

[42] P. O. M. Scokaert and J. B. Rawlings. Feasibility issues in linear model predictive control. *AIChE Journal*, 45(8):1649–1659, August 1999.

[43] S. Skogestad and M. Morari. Implications of large RGA elements on control performance. *Industrial and Engineering Chemistry Research*, 26:2323–2330, 1987.

[44] S. Skogestad and M. Morari. Robust performance of decentralized control systems by independent designs. *Automatica*, 25(1):119–125, 1989.

[45] S. Skogestad and I. Postlethwaite. *Multivariable Feedback Control. Analysis and Design.* John Wiley & Sons Ltd, Chichester, England, 1996.

[46] D. Sourlas, T. Edgar, and V. Manousiouthakis. Best achievable low order decentralized performance. In *Proceedings of the American Control Conference*, Baltimore, Maryland, June 1994.

[47] D. D. Sourlas and V. Manousiouthakis. Best achievable decentralized performance. *IEEE Transactions on Automatic Control*, 40(11):1858–1871, 1995.

[48] M. L. Tyler and M. Morari. Performance assessment for unstable and nonminimum-phase systems. Technical report, California Institute of Technology, 1995. IfA-Report no. 95-03.

[49] E. A. Wolff. *Studies on Control of Integrated Plants.* PhD thesis, Department of Chemical Engineering, University of Trondheim - NTH, 1994.

[50] E. Zafiriou and A. L. Marchal. Stability of SISO quadratic dynamic matrix control with hard output constraints. *AIChE Journal*, 37(10):1550–1560, October 1991.

[51] J. Ziegler and N. Nichols. Optimum settings for automatic controllers. *Trans. ASME*, 64:759–768, 1942.