The behavior of a MPC controller and the implementation in PRIMACS

Peeters, J.J.F.A.

Award date:
1995

NR-1906 (mei 1995)
Technische Universiteit Eindhoven
Faculteit Technische Natuurkunde
Vakgroep Systeem- en Regeltechniek

The behavior of a MPC controller and the implementation in PRIMACS

J.J.F.A. Peeters

Afstudeerverslag:

Afstudeerdocent:	Prof.dr.ir. J.J. Kok
Begeleiders:	Dr.ir. R.P.J. van der Linden
	Ir. W.J. Bouman

Summary

Model Predictive Control (MPC) is a control strategy introduced by Richalet in 1978. This sophisticated control strategy combines feedforward control with feedback control. The strategy is based on the prediction of the plant behavior by means of a model. One of the major advantages of MPC over other control strategies is the flexible and explicit manner it can account for constraints.

This graduation project had two objectives. The first objective was the analysis of the behavior of the MPC controller. The MPC controller has been compared to a LQG controller on a setpoint tracking problem. Some rules of thumb were found to tune the prediction and control horizon of the MPC controller. The offset found from the setpoint values can be explained from the fact only the first sample of the prediction horizon is updated directly with a measurement. Furthermore an example is given the way the MPC controller handles predicted constraints and the implication to the controlled outputs.

The second objective was to implement a MPC controller in a software environment PRIMACS. The software environment has been developed by TNO-TPD, in Delft. The MPC controller controls a simulated plant. In order to be able to use the MPC controller, various input parameters are necessary. The implementation and validation of these input parameters has been decoupled of the rest of the MPC control software. Further more two interfaces have been introduced, in order to improve the data transport. Also the concept of a scenario approach was introduced. This scenario approach gives the user the ability to manipulate the behavior of the plant and controller at any time during the simulation. As a consequence of the many simulations and the correction of the MPC controller software where necessary, the reliability of the MPC controller has been improved.

Ik wil iedereen bedanken die mij gedurende dit jaar
gesteund en geholpen heeft, met name mijn begeleiders
Ruud van der Linden en Wim Bouman en
in het bijzonder Sandra, Geert, Martin en natuurlijk mijn ouders.

Table of Contents

1 . Introduction

Control configurations are often used in the industry. A control configuration is called a perfect controller when it keeps the outputs of a process continuously on a desired setpoint level in the presence of disturbances and setpoint changes. The feedforward control configuration depicted in Figure 1 has the theoretical potential for perfect control. A feedforward control configuration measures the disturbances $\underline{d}$ directly to eliminate the disturbance impact on the controlled outputs. To design a controller a model is used, which describes the behavior of the process. The knowledge about the behavior can be obtained by performing experiments or making a theoretical analysis of the process. Feedforward control cannot compensate for model errors. The compensation of model errors can be done by receiving information from process measurements. This is called feedback control. A scheme of a feedback controller is depicted in Figure 2. The feedback control configuration reacts after it has detected a deviation from the desired setpoints.

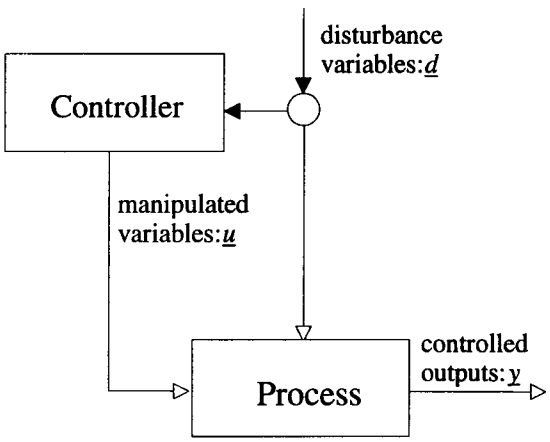


Figure 1: Scheme of a feedforward controller

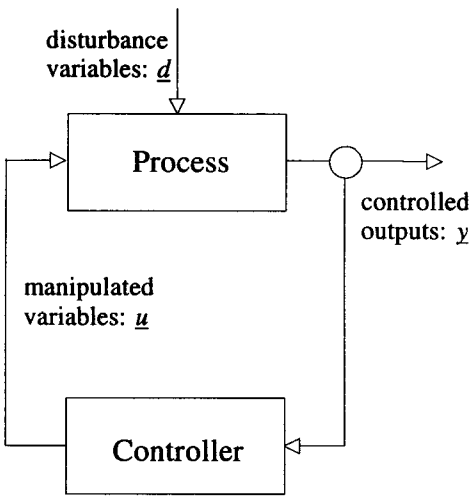


Figure 2: Scheme of a controller with a feedback loop

There are various types of controllers. The most frequently used controller is the Proportional Integral Derivative (PID) controller. A more sophisticated type of control, however, is performed by Model Predictive Control (MPC). In this controller feedforward control and feedback control are combined. MPC was first introduced by Richalet in 1978 [Ric78]. At that time the applications of MPC were limited by the computer power needed to compute the optimal control variables. Nowadays this computer power is not a limiting factor anymore and numerous applications are known with Model Predictive Control [Gar89] [Bal90] [Eat92].

The optimization of an industrial process to increase the profit rate will often lead to process operations at the intersection of constraints. MPC is a flexible controller which explicitly can account for constraints. This explains the popularity of the controller.

TNO-TPD in Delft is developing a software tool to be used by control engineers. This tool will be helpful for example to analyze data, to test various types of controllers on a process or to build models needed to control plants. The MPC controller is the controllers to be implemented in this software tool, during this graduation.

This report summarizes the graduation project. The first objective of this project was to implement a MPC controller in the software tool. The second objective was to show and analyze the behavior of the controller and to give some rules of thumb about the tuning of the controller. The graduation has been performed within the System & Control group of the Faculty of Technical Physics of the Eindhoven University of Technology in co-orperation with TNO-TPD, Delft.

In chapter 2 the general principle of Model Predictive Control is explained. Also the control algorithm, the estimator and the Logical Controller are clarified.

In chapter 3 the contributions to the software developed by TNO-TPD are described and explained. Besides the rewriting of the software and the validation of the controller two new features were added. One new feature was a data interface and the other feature was the validation and implementation of new parameter input on basis of a scenario approach.

In chapter 4 the model of a Fluidized Catalytic Cracker (FCC) is described. This model has been used to test the MPC controller. Furthermore a comparison is made between the non-linear FCC plant and the linear state space model used by the MPC controller to control the FCC process.

In chapter 5 the behavior of the process and the plant in different situations are shown and explained. Furthermore in this chapter the MPC controller is compared with a LQG controller and some rules of thumb are found concerning the tuning of the MPC controller.

In chapter 6 conclusions and recommendations are given.

2 . Theory

2.1 . The principle of Model Predictive Control

The principle of MPC is easy to comprehend. At time k first the controller predicts what will happen to the process outputs $\underline{y}$ over a so called prediction horizon p, while the manipulated variables $\underline{u}$ are kept constant at their current value. Usually the plant outputs $\underline{y}$ do not agree with the wanted setpoint values. In Figure 3 is depicted that by varying the manipulated variables $\underline{u}$ around the fixed values, the control sequence can be calculated for which the wanted setpoint of the outputs $\underline{y}$ are met best. The calculation is done with the use of the process model. The control algorithm is allowed to vary the manipulated variables over a control horizon m. After the best control sequence has been found, the first of this sequence is used as the new input to the process. Because a discrete control loop is considered the control signal is kept constant over a sample time $\Delta\tau$.

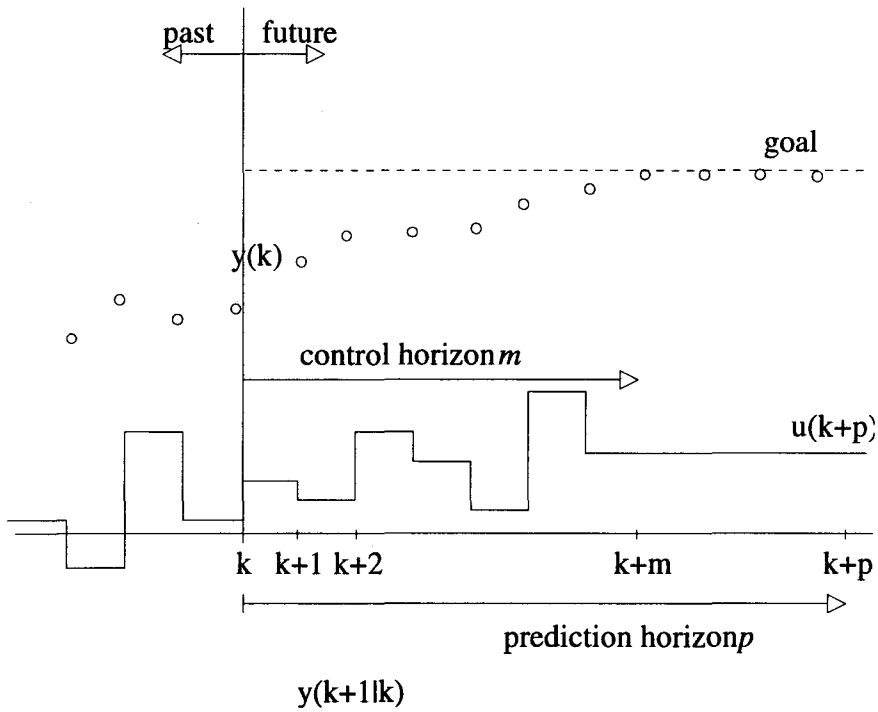


Figure 3: Finding the optimal control sequence u(k+i) in order to reach the wanted outputs.

In the feedback loop, measured plant outputs give information on the state of the process every sample. When calculating the sequence of optimal manipulated variables to control the process, the latest information from the process should be used. Therefore every sample the manipulated variables are calculated again. This means that the start point of the prediction and control horizon is moved from t = k to t = k+1 (see Figure 3). This is called the moving horizon principle. The model used to predict the future state variables, is a state space model.

2.2. Estimator

As mentioned above, the MPC controller uses measured information to reach a better solution of the control problem. Before the actual control sequence starts, the model is brought in the same state as the plant. This is done in the estimator. Filtered measured data give some information on the state of the plant. The state variables $\underline{x}$ are updated once every sample time on basis of both the measured and filtered outputs and the model (from now on the vectors are written without underscore). The state variables $x(k+1)$ are influenced by the old state variables $x(k)$ and the deterministic manipulated variables $u(k)$ and 2 modeled disturbances:

- $d(k)$: measured disturbances.
- $z(k)$: unmeasured disturbances.

To estimate the new state variables the following state space model can be used. In appendix A the signification of the symbols are explained.

$$\begin{aligned} x(k+1) &= \Phi x(k) + \Gamma_u u(k) + \Gamma_d d(k) + \Gamma_z z(k) + \Omega v(k) \\ y_m(k) &= C_m x(k) + D_{mu} u(k) + D_{md} d(k) + w(k) \end{aligned} \tag{2.1}$$

where $x(k)$ is the state vector, $v(k)$ is the model noise vector, $w(k)$ is the measurement noise vector, Φ is the state matrix, Γ_u is the manipulated variable matrix, Γ_d is the measured disturbance matrix, Γ_z is the unmeasured disturbance matrix, Ω is model noise vector, C_m is the output matrix , D_{mu} is the manipulated variable output matrix, D_{md} is the disturbance variable output matrix. C_m, D_{mu}, D_{md} and $w(k)$ are appropriately adapted to the measured outputs y_m.

The unknown disturbances are modeled as follows:

$$z(k+1) = z(k) + v_z(k) \tag{2.2}$$

with the $v_z(\mathrm{k})$ - elements are independent white noise signals with known variances. When the unmeasured disturbances are modeled like in equation (2.2), this is comparable to an integrating action. The addition of an unmodeled disturbance is used to compensate for the mismatches between the model and the plant at steady state.

Using equation (2.2), the estimator problem can be reformulated. The state vector x can be augmented with the vector z:

$$\begin{aligned} x_a(k+1) &= \Phi_a x_a(k) + \Gamma_u u(k) + \Gamma_d d(k) + \Omega_a v_a(k) \\ y_m(k) &= C_{ma} x_a(k) + D_{mu} u(k) + D_{md} d(k) + w(k) \end{aligned} \tag{2.3}$$

with the subscript a as symbol for the augmented state.

At time k the MPC controller calculates the augmented state vector $x_a(k+1)$ and sends $u(k)$ to the process. Then the controller measures the output variables $y_m(k+1)$. The state variables $x_a(k+1)$ are updated with these measurements, thus also $z(k+1)$:

$$\hat{x}_a(k+1) = \bar{x}_a(k+1) + K\bar{\varepsilon}(k+1) \tag{2.4}$$

with:

$$\bar{\varepsilon}(k+1) = y_m(k+1) - \bar{y}_m(k+1)$$

where $\bar{x}_a(k+1)$ are the state variables before the measurement update, $\hat{x}_a(k+1)$ are the state variables after the measurement update, K is the Kalman filter, $y_m(k+1)$ are the measured outputs at $k+1$ and $\bar{y}_m(k+1)$ are the estimated outputs at $k+1$.

The variable $\bar{y}_m(k+1)$ is calculated with the one step ahead prediction model:

$$\begin{aligned} \bar{x}_a(k+1) &= \Phi_a \hat{x}_a(k) + \Gamma_u u(k) + \Gamma_d d(k) \\ \bar{y}_m(k+1) &= C_{ma}\bar{x}_a(k+1) + D_{mu}u(k+1) + D_{md}d(k+1) \end{aligned} \tag{2.5}$$

Because the values $u(k+1)$ are yet to be calculated the last calculated values $u(k)$ are used. $u(k)$ means the values of the manipulated variables on $t=k$. The values are kept constant until $t=k+1$. Then the new values for the manipulated variables u are calculated.

2.2.1. The discrete Kalman filter K

In this section the determination is given of the Kalman filter introduced in equation (2.4). The Kalman filter K is related to the ratio of model noise and measurement noise. Both these noises are modeled as non zero white noise. The Kalman filter [Bro83] is used to improve a prior estimation of the state variables by the model. Before the working of the filter is explained, first some definitions are given in equation (2.6) and equation (2.7).

Without loss of generality, equation (2.1) can be altered in the following state space model:

$$\begin{aligned} x_{k+1} &= \Phi_k x_k + v_k \\ y_k &= C_k x_k + w_k \end{aligned} \tag{2.6}$$

where

$x_k = (n \times 1)$ process state vector at time t_k
$\Phi_k = (n \times n)$ matrix relating x_k to x_{k+1}
$v_k = (n \times 1)$ model error vector, assumed to be white noised
$y_k = (m \times 1)$ measurement vector at time t_k
$C_k = (m \times n)$ matrix, describing the relation between the measurement and x_k at time t_k
$w_k = (m \times 1)$ measurement error vector.

Both the model error vector and the measurement error vector are related to one another by the following covariance matrices:

$$
\begin{aligned}
E(v_k v_i^T) &= \begin{cases} Q_k, & i = k \\ 0, & i \neq k \end{cases} \\
E(w_k w_i^T) &= \begin{cases} R_k, & i = k \\ 0, & i \neq k \end{cases} \\
E(v_k w_i^T) &= 0, \quad \forall i, k
\end{aligned}
\tag{2.7}
$$

where Q_k is the model noise covariance matrix and R_k is the measurement noise covariance matrix. Both matrices are diagonal.

Now the error vector e_k^- is introduced:

$$
e_k^- = x_k - \hat{x}_k^- \tag{2.8}
$$

where x_k is the true state vector, $\hat{x}_k^-$ is the best estimation of the state vector before measurement. The associated covariance matrix P_k^- of error vector e_k^- is:

$$
P_k^- = E\left[e_k^- e_k^{-T}\right] = E\left[(x_k - \hat{x}_k^-)(x_k - \hat{x}_k^-)^T\right] \tag{2.9}
$$

P_k^- expresses the mean square deviation of the true state vector and the state vector before measurement. The best estimation of the state vector $\hat{x}_k^-$ can be improved with measurement data. This leads to an updated state variable $\hat{x}_k$:

$$
\hat{x}_k = \hat{x}_k^- + K_k(y_k - C_k \hat{x}_k^-) \tag{2.10}
$$

where $\hat{x}_k$ is the estimate update state vector and K_k is the blending factor.

The associated covariance matrix P_k of the error vector e_k expresses the mean square deviation of the true state vector and the state vector after the measurement update. Noting the a priori estimation error e_k^- is uncorrelated with the measurement vector w_k, then it is possible to rewrite P_k:

$$
\begin{aligned}
P_k &= E\left[e_k e_k^T\right] = E\left[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T\right] \\
&= E\left\{\left[\left(x_k - \hat{x}_k^-\right) - K_k(C_k x_k + w_k - C_k \hat{x}_k^-)\right]\left[\left(x_k - \hat{x}_k^-\right) - K_k(C_k x_k + w_k - C_k \hat{x}_k^-)\right]^T\right\} \\
&= (I - K_k C_k) P_k^- (I - K_k C_k)^T + K_k R_k K_k^T
\end{aligned}
\tag{2.11}
$$

where I is the unity matrix.

To find the best new states update, the associated covariance matrix P_k must be minimized. Therefore a particular K_k has to be found because this is the only matrix that can be influenced. By straightforward calculations the optimal Kalman filter K_k^{opt} and covariance matrix P_k can be expressed as a function of P_k^- and K_k^{opt}:

$$\begin{aligned} K_k^{opt} &= P_k^- C_k^T (C_k P_k^- C_k^T + R_k)^{-1} \Rightarrow \\ P_k &= (I - K_k^{opt} C_k) P_k^- \end{aligned} \tag{2.12}$$

K_k^{opt} is called the Kalman gain matrix for time t_k.

At $t=k+1$ the best estimate of the states x_{k+1}^- and the error vector e_{k+1}^- are found using equation (2.13):

$$\begin{aligned} x_{k+1}^- &= \Phi_k \hat{x}_k \\ e_{k+1}^- &= x_{k+1} - \hat{x}_{k+1} = (\Phi_k x_k + v_k) - \Phi_k \hat{x}_k = \Phi_k e_k + v_k \end{aligned} \tag{2.13}$$

Because e_k and v_k are uncorrelated, P_{k+1}^- can be expressed as, with inserting equation (2.13):

$$P_{k+1}^- = E[e_{k+1}^- e_{k+1}^{-T}] = \Phi_k P_k \Phi_k^T + Q_k \tag{2.14}$$

Because in this particular case the used model does not change in time, the time independent Kalman gain matrix can be determined. In the steady state situation $P_{k+1}^- = P_k^-$. The Riccati equation is deduced by inserting equation (2.11) into equation (2.13). This equation can be solved with the Schur method:

$$\begin{aligned} & P_{k+1}^- - P_k^- = 0 \Rightarrow \\ & \Phi_k P_k^- \Phi_k^T - \Phi_k P_k^- C_k^T (C_k P_k^- C_k^T + R_k)^{-1} C_k P_k^- \Phi_k - P_k^- + Q_k = 0 \end{aligned} \tag{2.15}$$

The matrix found for P_k^- is inserted in equation (2.12) to find the Kalman gain filter K_k^{opt}.

2.3. Prediction

2.3.1. Introduction

In the estimator values for the state variables x are found, which represent the state of the controlled process better. These values are a good starting point for the prediction of the future states of the process. With this future prediction the optimal control moves can be found as stated in section 2.1. In order to find this sequence of control moves, the prediction algorithm is separated in two parts. The first part predicts the future state variables by setting the manipulated variables on the current values. This is an open-loop response. The second part predicts the state variables (and the process outputs) by varying the manipulated variables. The manipulated variables are optimized to meet the stated goals. This part is the actual optimization and is treated in the next section.

2.3.2. Prediction Method

The predictions are based on the following equation (see equation (2.1)):

$$x(k+i+1) = \Phi x(k+i) + \Gamma_u u(k+i) + \Gamma_d d(k+i) + \Gamma_z z(k+i) \tag{2.16}$$

The state variables from k to $k+p$ can be expressed in variables all known at time k:

- the state variables at sample time k: $\hat{x}(k)$.
- the control moves at sample time k to $k+m$: $\Delta u(k+i)$, *with* $0 \le i \le m$.
- the predicted disturbances at sample time k to $k+s$: $\Delta d(k+i)$, *with* $0 \le i \le s$.
- the estimated unmeasured disturbance $\hat{z}(k)$, the measured disturbance $d(k)$ at sample time k and the current control move $u(k\text{-}1)$.

By using equation (2.16) the state variables and this list stated above, $x(k+i)$ can be written as:

$$\begin{aligned} x(k+i) = \Phi^i \hat{x}(k) + \sum_{i=1}^{p} \Phi^{i-1}(\Gamma_u u(k-1) + \Gamma_d d(k) + \Gamma_z \hat{z}(k)) + \sum_{i=1}^{s} \Phi^{i-1}\Gamma_d \Delta d(k+i) + \\ \sum_{i=1}^{m} \Phi^{i-1}\Gamma_u \Delta u(k+i) + \sum_{i=m+1}^{p} \Phi^{i-1}\Gamma_u \Delta u(k-m+i), \quad for\ i > m \end{aligned} \tag{2.17}$$

where $\Phi^i = \Phi^{i-1} \cdot \Phi$.

This formula can be written in a matrix and vector notation over the complete prediction horizon p.

$$\begin{aligned} x^p(k+1) = \Phi^p \hat{x}(k) + S(\Gamma_u u(k-1) + \Gamma_d\ d(k) + \Gamma_z \hat{z}(k)) + \\ S_d L_d \Delta \bar{d}(k+1) + S_u L_u \Delta u^m(k) \end{aligned} \tag{2.18}$$

where $x^p(k+1)$ is a super vector, which contains the state vectors $x(k+i)$ over the complete prediction horizon and Δu^m is a super vector, which contains the manipulated variables $\Delta u(k-m+i)$ over the complete control horizon m. S, S_d, and S_u are super matrices, which contain the summons of the state matrix Φ. L_u and L_d are also super matrices, which hold the manipulated variable matrix and disturbance variable matrix, respectively.

Equation (2.18) can be separated in the two parts mentioned above.

$$x^p(k+1) = x^p_{\Delta u=0}(k+1) + x^p_{\Delta u}(k+1) \tag{2.19}$$

with:

$$x^p_{\Delta u=0}(k+1) = \Phi^p \hat{x}(k) + S(\Gamma_u u(k-1) + \Gamma_d d(k) + \Gamma_z \hat{z}(k)) + S_d L_d \Delta d(k+1) \tag{2.20}$$

and

$$x^p_{\Delta u}(k+1) = S_u L_u \Delta u^m(k) \tag{2.21}$$

where $\Delta u^m(k)$ is calculated with the optimization algorithm.

The MPC controller as implemented in the software tool, uses the feedback measurements to correct the state variables of the next sample. With these updated variables and the plant model the state variables are calculated over the prediction horizon p. Morari [Mor91] uses a different approach. The feedback measurements are used to correct all the output variables over the prediction horizon p afterwards. This is done by adding a constant bias to all the prediction samples:

$$y^N(k+1) = y^N(k) + T\{y_m(k) - y(k)\} \tag{2.22}$$

where $y^N(k)$ are the output variables over the prediction horizon N at $t=k$, T is the noise filter, $y_m(k)$ are the measured values of the outputs at $t=k$ and $y(k)$ is the prediction of the outputs for $t=k$.

Morari uses step response models. So instead of the state variables the output variables are predicted. This does not differ from the method proposed here, because the predicted future outputs are determined with the use of the state space model:

$$\begin{aligned} y^p_{\Delta u=0}(k+1) &= C^p x^p_{\Delta u=0}(k+1) + D_u^{p+1} u^{p+1}(k-1) + D_d^p \bar{d}^p(k+1) \\ y^p_{\Delta u}(k+1) &= (C^p S_u L_u + D_u^{p+1} L_u)\Delta u^m \equiv A \Delta u^m \end{aligned} \tag{2.23}$$

where C^p, D_u^{p+1} and D_d^{p+1} are super matrices and where A is introduced to simplify the equation.

2.4. Optimization

2.4.1. Introduction

The objective of a control engineer is that a process behaves in a certain way. The process industry often demands optimal economical behavior, like maximum profit or minimal costs. In research a wanted behavior can be an as pure as possible output. The process, its inputs or both often are limited in their manipulation range, e.g. finite dimensions of the flow pipe, or when operational safety has to be guaranteed. If many criteria have to be optimized at the same time, the problem is nearly infeasible. Therefore, only the most important criterion (usual the economical one) is optimized while the others are treated as constraints.

2.4.2. The algorithm

The criterion to be optimized is often transferred to a setpoint objective function. In this objective function the goal of the optimization is expressed in setpoints. The objective function has the following form:

$$\min_{\Delta u^m(k)} J = \sum_{i=0}^{p} \|\Gamma_y (y(k+i) - r(k+i)\|^2 + \sum_{i=0}^{m} \|\Gamma_u \Delta u(k+i)\|^2 \qquad (2.24)$$

where $r(k+i)$ is a vector, which contains the desired setpoint values of the process outputs. Γ_u and Γ_y are weight matrices and they express the relevancy of the corresponding objectives.

Also constraints have to be taken into account. On the input of the plant a control variable can be restricted between a minimum and maximum value. On the output side the output values have to stay between two arbitrary values. Three types of constraints can be distinguished:

- manipulated variable constraints: hard constraints on the inputs u, for example maximum inlet pressure.
- manipulated variable rate constraints: hard constraints on the change input value.
- output variable constraints: hard and soft constrains are imposed to prevent the process from unwanted behaviour like overshoot.

Hard constraints can not be released. Soft constraints can be released, in order to find a feasible solution for the control problem.

The objective function in equation (2.24) is not appropriate to find the optimal sequence of control moves because the first term on the right hand site does not include $\Delta u^m(k)$ explicit.

Rewriting this function with equation (2.23) results in equation(2.25).

$$\min_{\Delta u^m(k)} J = \sum_{i=0}^{p} \left\| \Gamma_y (E(k+i) - A\Delta u(k+i) \right\|^2 + \sum_{i=0}^{m} \left\| \Gamma_u \Delta u^m(k) \right\|^2 \qquad (2.25)$$

$$\textit{subject to} \quad C_{\Delta u_m} \Delta u^m(k) \geq M(k)$$

with

$$E\ (k+i) = y_{\Delta u=0}(k+i) - r(k+i)$$

and

$$M(k) = \begin{bmatrix} \Delta u_{\max}(k) \\ \vdots \\ \Delta u_{\max}(k+m-1) \\ -\Delta u_{\max}(k) \\ \vdots \\ -\Delta u_{\max}(k+m-1) \\ u_{low}(k) - u_{meas}(k-1) \\ \vdots \\ u_{low}(k+m-1) - u_{meas}(k-1) \\ -u_{high}(k) - u_{meas}(k-1) \\ \vdots \\ -u_{high}(k+m-1) - u_{meas}(k-1) \\ y_{low}(k) - y_{\Delta u=0}(k) \\ \vdots \\ y_{low}(k+p) - y_{\Delta u=0}(k) \\ -y_{high}(k) - y_{\Delta u=0}(k) \\ \vdots \\ -y_{high}(k+p) - y_{\Delta u=0}(k) \end{bmatrix}, \quad C_{\Delta u_m} = \begin{bmatrix} I \\ -I \\ I \\ -I \\ A \\ -A \end{bmatrix}$$

As mentioned in section 2.1, the manipulated variables are varied over a control horizon m around the current control value. The $M(k)$ - vector expresses the range over which the manipulated variables $\Delta u^m(k)$ are allowed to be varied, with $\Delta u_{\max}$ is manipulated variable rate constraint, u_{low} and u_{high} are manipulated variable constraints and y_{low} and y_{high} are output variable constraints.

The problems in equation (2.24) and equation (2.25) are known as Quadratic Programming (QP) problems. First the unconstrained QP problem is solved, in order to give a better explanation of the calculation of the solution of the QP problem with constraints. Because this unconstrained QP problem costs little computer time to solve, it is solved every simulation step. When the solution of the unconstrained problem is known, it is checked if any constraints are violated. If this is the case, the constrained QP problem has to be solved.

The unconstrained QP problem can be solved analytically. This can easily be accomplished by introducing the residue vector ρ in the following way:

$$\rho = b - Fx = \begin{bmatrix} \Gamma_y \\ 0 \end{bmatrix} - \begin{bmatrix} \Gamma_y A \\ \Gamma_u \end{bmatrix} \Delta u^m(k) \tag{2.26}$$

The minimization of the 2-norm of the residues ρ is now straightforward.

$$\min_x \rho^T \rho = \min_x (b - Fx)^T (b - Fx) \tag{2.27}$$

Solving this analytically the first order derivative has to be 0:

$$\frac{d(\rho^T \rho)}{dx} = -2F^T(b - Fx) = 0 \Rightarrow x = (F^T F)^{-1} F^T b \tag{2.28}$$

For the extreme to be a minimum the second order derivative has to be semi-positive definite. This leads in the unconstrained QP problem to the following solution:

$$\Delta u^m(k) = (A^T \Gamma_y^T \Gamma_y A + \Gamma_u^T \Gamma_u)^{-1} A^T \Gamma_y^T \Gamma_y E^p(k+1) \tag{2.29}$$

The constrained QP problem can be solved by optimization routines explained in Appendix C.

2.5. Logical Controller

In the previous sections the basic MPC controller and the estimator have been explained. The basic controller and the estimator can be called the control calculation part. Around this part a Logical Controller unit has been added. This Logical Controller takes over various tasks an operator would normally do:

- to test if the measured outputs y_m still can be measured.
- to test if a manipulated variables u still can be used.
- to determine which constraints can temporarily be released.
- to determine if an extra manipulated variable can be used.
- to determine which setpoints have to be controlled.

The purpose of a Logical Controller is to increase the robustness of the controller in a real process situation. To find the optimal manipulated variables to implement, every sample the following sequence of tasks is done. Point 1, 3, 4, 6, 8, 9, 10 are done in the Logical Controller unit. Point 2, 5, 7 are done the control calculation part.

1. Test the conditions to calculate a new Kalman gain matrix.
2. Compute the predictions of variables over the entire prediction horizon, keeping the manipulated variables constant at their current value.
3. Detect if the operability of manipulated variables are changed and take the appropriate actions if necessary.
4. Select the complete set of constraints.
5. Compute the unconstrained MPC solution. If the problem is unconstrained, the control calculation part is finished.
6. Test if any of the constraints are violated in the unconstrained case. If not, the control calculation part is finished.
7. Use the QP optimization routine and test if an optimal feasible solution has been found. If this is true, the control calculation part is finished
8. Detect if there are manipulated variables labeled for restricted use (see below), that are not yet used. If true, add the one with the lowest priority to the current manipulated variables and return to 3.
9. Test the constraint priority list (see below) and remove the one with the lowest priority and go back to 6.
10. If no feasible solution can be found, return an error message.

Regarding point 8, when a manipulated variable is labeled for restricted use, this variable can be used only in exceptional occasions. There are several reasons to label a variable for restricted use. An economical reason, the variable is expensive to use or a safety reason, to use the variable for a longer time endangers the safety of the plant. If more manipulated variables are labeled for restricted use a priority is added. The one with the lowest priority is used the first when necessary.

Regarding point 9, there are also many reasons to put constraints on input and output variables. Sometimes these constraints are inevitable: "hard constraints" (see section 2.4) and other constraints can be released for a shorter or longer period: "soft constraints". For example: often the outputs related to quality must be between an under and upper constraint in order to satisfy the specifications of the product. If for some cause one of the constraints is about to be violated, the constraints on one of the manipulated variables can temporarily be released.

The priority list both used for the manipulated variable for restricted use and soft constraints are user defined.

3 . Simulation environment

3.1 . Introduction

The Model Predictive Controller has been implemented in the software environment PRIMACS developed by TNO-TPD, Delft. In this environment different applications can be designed to help the control engineer, for example to test new control techniques, to model plants or to analyze data. The communication between the user and PRIMACS runs via question pages, where the wanted information is asked to run an application. It must be possible to apply different control techniques to different processes/plants.

During this graduation project the MPC controller implemented here is an extension of the one developed by S. Strand as a result of the PROFIT project [Pro92]. This controller, including the logic control features as described in chapter 2, was written in the computer language C. The parameter inputs were read through files and the model used to control the plant was a state space model. The outputs of both the plant and the controller were written to files too. For the visualization of the data the software program Matlab™ was used. Before the controller was redesigned for the use in PRIMACS, first the controller had to be validated.

The redesign of the S.Strand software had the following implications:

- using the PRIMACS matrix structures and its dbase functionality.
- during the simulation the user must be able to change parameters of the controller at any time. The handling of the parameter changes is performed at a later time point (see section 3.2).
- the implementation of data interfaces between the outputs and inputs of the controller.
- porting the software from C to C++ and going from operating system VAX/VMS to DOS.
- starting to build a simulation environment on basis of a scenario approach (yet to be explained) where controller and/or plant can be analyzed.

3.2 . Parameter input

To test a controller, often a simulation of a plant is used which has to be controlled by the controller. In this way the behavior of the controller in different situations can be analyzed fast and without any danger. The sequence of different situations can be seen as a scenario of what the controller faces. In the simulation environment build to test the MPC controller, the user must be able to add or alter situations at any time, including parameter change to the controller.

From the MPC controller side of view the input parameters are divided in 5 logical structures:

- the model structure contains information on the state space model.
- the control structure contains information necessary to manipulate the prediction and control horizon.
- the input structure contains information on the way the manipulated variables should be used.
- the output structure contains information in which manner the outputs have to be treated.
- the estim structure contains information on the estimated model noise

To test if the new parameters are consistent with the other input data, the validation is done in 2 steps:

1. the altered data has to be consistent within the structure, this is done on-line.
2. the altered data has to be consistent with the other data in the buffer.

With respect to the first step the new input data itself must be correct (e.g. no negative prediction horizon). With respect to the second step it has been chosen to validate the data in order of implementation, because this leaves the user the most time to change the non consistent parameters

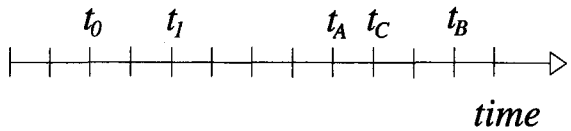


Figure 4 :Time points of user inserting the data and time points of implementing in controller.

Consider the time scale given in Figure 4. At t_0 first the new estimated model noise data is validated for time point t_A , followed by the validation of the new measurement standard deviations for time point t_B. At time t_1 the new state space model is validated with the already in use parameters and the ones to be implemented at t_A. If for example a model is used with more disturbance observers than outputs to be measured, the validation is incomplete. Because it is chosen to validate in order of implementation, the new measurement standard deviations to be implemented at t_B are revalidated.

To keep the administration open and clear an "evaluation" matrix EM is introduced (see Table 1). This is a 5 x 3 matrix. The rows correspondent to the 5 logical structures. The first column contains a kind of Boolean elements.

If an element is set TRUE, new non validated parameters are in the PRIMACS dbase. An element is set SPECIAL_CHANGE, if the new parameters can be directly implemented on time t_z (moment on which the user inserts this data). The SPECIAL_CHANGE data has to do with changes in constraints or setpoints. The sooner the MPC controller has information on the changes, the sooner it can take this new changes into account. This is one of the major advantages of MPC.

If in the second column a Boolean element is set TRUE the new data have been validated. Both columns are set FALSE when the MPC controller uses the data. The altered parameters have to be implemented in the program at an absolute time x. These absolute times are kept in the third column. As mentioned before due to the scenario approach the various different user's inputs are validated in order of the lowest time point x.

Groups	*New parameters are available, but not yet implemented*	*New parameters have been validated, but not yet implemented*	*Time on which the new parameters must be implemented*
Model	FALSE	FALSE	
Input	TRUE	FALSE	75
Output	SPECIAL_CHANGE	FALSE	40
Control	FALSE	TRUE	60
Estim	TRUE	FALSE	90

Table 1: The evaluation matrix EM with an input example.

If there is new input data, there are three possible cases: this data must be implemented on a time point earlier than some of the already altered but not yet implemented data, on a later time point or on the same time point. This leads to the following flowchart (Figure 5).

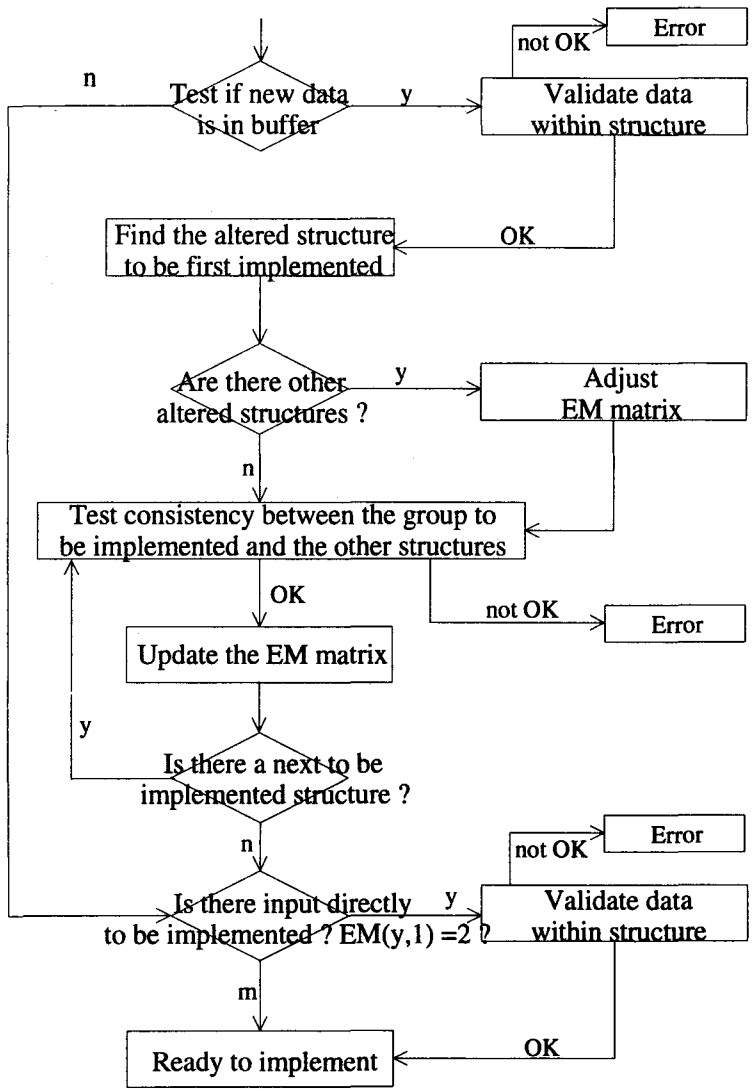


Figure 5: Flowchart of the data consistency test.

After a good data consistency test, all the parameters are consistent with each other over the simulation time. It should be noticed that the parameters which are already used by the MPC controller, never have to be modified due to other altered input parameters.

3.3 . Data interfaces

As mentioned before the plants/processes and the controllers are interchangeable in the ideal case. An I/O standard has to be developed to make this type of linking possible. Another reason to design an interface is to analyze the behavior of the controller in different situations e.g. when measurement noise is added, known and unknown disturbances are used as inputs to the plant. To simulate this type of situations the data interfaces are also used. In Figure 6 the data interfaces are shown as a part of the control loop.

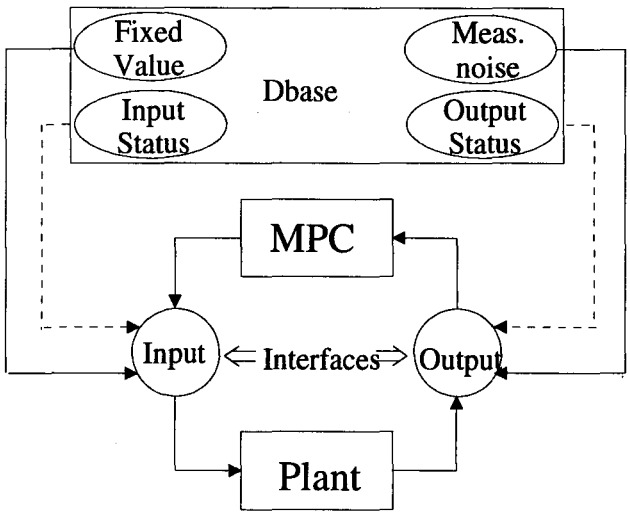


Figure 6: The data interfaces and their connection with the dbase.

The options on the input side of the plant are (input status):

- normal mode: the calculated input is without changes put on the plant input.
- steady mode: a fixed value of a manipulated variables is put on the plant input.
- disturbance mode: implementation of disturbances on the plant input.
- another manipulated variable behaviour: defined and programmed by the user (not yet implemented).

If the user chooses a manipulated variable to malfunction, the user can choose if it is known to the controller if the manipulated variable is defect.

On the output side of the plant the first choice is which of the plant outputs are measured by the controller. The data interface can modify a plant output in four different ways (output status):

- normal mode: plant output is not changed.
- noise mode: white noise is put on the outputs, the measurement standard deviations can be defined by the user.
- offset mode: an arbitrary offset is added on the plant output.
- other: plant output behaviour defined by the user.

Like in the previous case the user can choose, if it is known to the controller if an output variable measurement is defect.

4 . Fluidized Catalytic Cracker

4.1 . Introduction

One of the methods used to upgrade the less valuable petroleum to gasoline, is Fluidized Catalytic Cracking (FCC). In this process catalyst is used to refine the petroleum. Because of the large volume of oil refined every day, even a small increase in efficiency pays important dividends to energy savings and profits. Therefore a better control of the FCC plants is an important objective. A mathematical model of this plant [Lee85][McF90] is used to test the MPC controller in this project. The most important problem is the deactivation of the catalyst because coke is deposited on the catalyst surface. This problem can elegantly be overcome when a regenerator is used, as will be explained in the next section.

The FCC process can be divided in 3 components:

- the riser
- the separator
- the regenerator

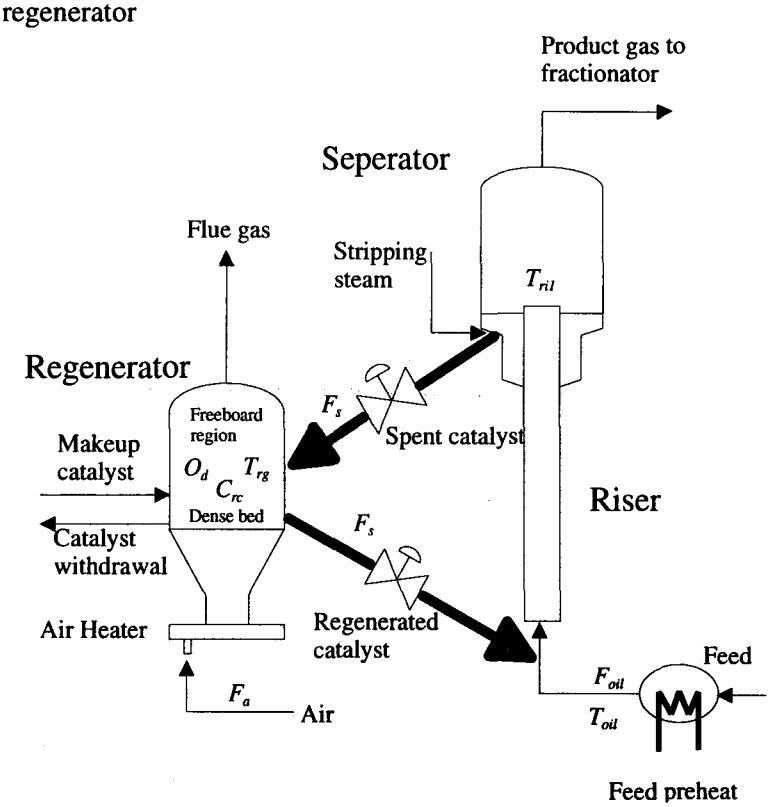


Figure 7: The three components of the FCC plant; the regenerator, the separator and the riser.

In Figure 7 a schematic diagram of the plant is shown. The oil feed mixes with the catalyst in the riser. The oil is "cracked" due to the zeolite catalyst. In the separator the cracked products are separated into gasoline, light gasses and heavier products. The spent catalyst is fed back to the regenerator where deposited coke is burnt off. The regenerated catalyst goes back into the riser carrying sufficient heat required to continue the endothermic cracking reactions. Besides the reversible catalyst deactivation also irreversible deactivation takes place. A low feed of new catalyst is fed to the regenerator. In the next sections the components of the plant will be discussed more in detail.

4.1.1. The riser

Preheated heavy oil is fed to the riser through nozzles. In the nozzles droplets are generated smaller, which are smaller than 100 μm in diameter. Through this process the oil is uniformly spread over the riser such that the catalyst can come easily into contact with the oil. The ratio between the catalyst particles and the oil particles varies between the 6:1 and 12:1. The temperature of the oil is in the range from 500 K to 900 K and its residence time is between the 2 and 8 seconds. Because the riser contains several ten thousands of different products, it is impossible to identify them all separately in a kinetic model. Therefore the lumped approach is used to tackle the problem of the large number of products. Only 4 products are taken into account for the kinetic model. These products are gasoline, gas oil, coke and gas(C_1-C_4). The rate constants used in this model are mostly company secrets. Therefore some typical numerical values of the parameters are used [Lee 85].

4.1.2. The separator

In the separator the produced gasses are separated from the catalyst. The separators used nowadays perform so well that they can be modeled as ideal separators. No reactions take place in the separator. It is only a capacity for the accumulation of coke and heat on the catalyst [Lju90].

4.1.3. The regenerator

The largest part of the catalyst is found in the regenerator and that is why the regenerator dominates the characteristic response of the total process. The coke has to be burnt off of the catalyst surface in order to reactivate the catalyst. This is done by feeding air to the dense bed (see Figure 7). Not only coke deactivates the catalyst, but also metallic particles and high temperatures destroy the crystalline structure of the catalyst. Steam accelerates this destruction process. The latter examples of deactivation are irreversible.

Here coke is defined with the chemical formula CH_n with n between 0.4 and 2.0, and is burnt to CO, CO_2 and H_2O. These processes take place in the dense bed. The reactions in the freeboard (see Figure 7) are mainly for further oxidation of CO to CO_2 with the O_2, which did not react with the coke. It is assumed that the solids and the gasses are homogeneous divided in the regenerator. The model is semi-empirical and it is assumed that the deactivation of the catalyst goes linear with the coke concentration on the catalyst. To take the temperature variations into account a time dependent function is added.

The model has 3 time constants of which the largest (+/- 65 min) hangs together with the coke on the regenerated catalyst [Lju90]. The smallest (+/- 0.008min) strongly depends on the oxygen balance of the catalyst, this time constant is set equal to a step in the used model. Fictive air accumulation in the regenerator is used to remove the model stiffness.

4.2 . Plant inputs and outputs

The standard choice of the manipulated variables to control the FCC plant are:

- F_a: air feed rate (kg/min)
- F_s: recirculation rate of the catalyst (kg/min)

The other input variables are mostly determined by higher optimizations and can be seen as disturbances:

- T_{oil}: temperature of the oil feed (K)
- F_{oil}: oil feed rate to the riser (kg/min)
- T_{air}: inlet temperature of air to the regenerator (K)

The changes in feed quality and catalyst properties can also been seen as disturbances. The non linearities and strong interactions between the FCC variables lead to a difficult control problem.

The controlled and measured outputs are:

- T_{ril}: temperature in the riser (K)
- O_d: oxygen fraction in the regenerator (mole frac.)
- T_{rg}: temperature in the regenerator (K)

T_{ril} gives information on the components of the refined oil [McF90]. The setpoint of T_{ril} can be set to the level economically most profitable level at that time. The temperature of T_{ril} is directly coupled to T_{rg}. T_{rg} gives information on the amount of energy flowing from the regenerator to the riser, necessary for the endothermically reaction of catalyst and petroleum. The higher the temperature T_{rg} the more catalyst becomes irreversibly inactive. Thus from an economical point of view this temperature has to be kept as low as possible.

4.3 . Plant simulations

In order to get a better understanding of the plant behavior, a qualitative explanation of the step responses on F_a and F_s is given.

4.3.1. Step responses

Figure 8 shows the 3 outputs of the FCC plant when a step of manipulated variable F_a is made from 1521.1 kg/min to 1421.1 kg/min and F_s stays at the working point value of 17640 kg/min.

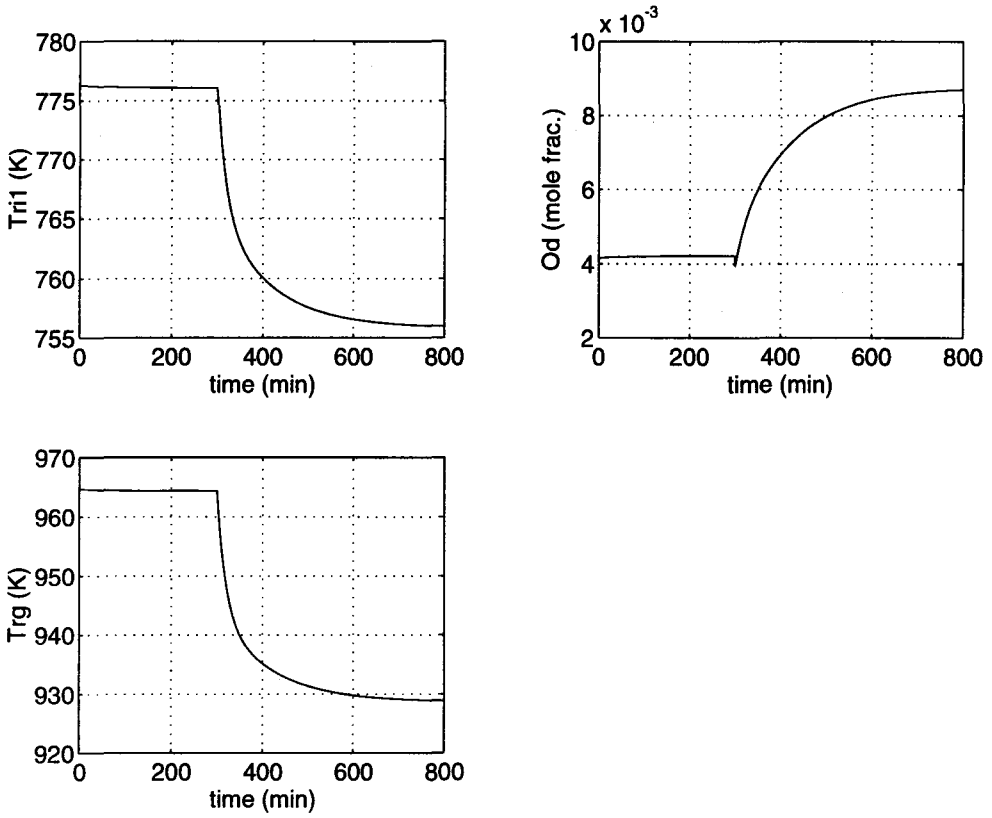


Figure 8: Step response on t=300 min. F_a=1421.1 kg/min and F_s at working point value.

Because of the decreasing of F_a the oxygen fraction O_d decreases. If less oxygen is available, the combustion will decrease, and therefore a temperature drop of T_{rg} takes place. The lower temperature contributes to an even less combustion and again less oxygen is needed (O_d rises).

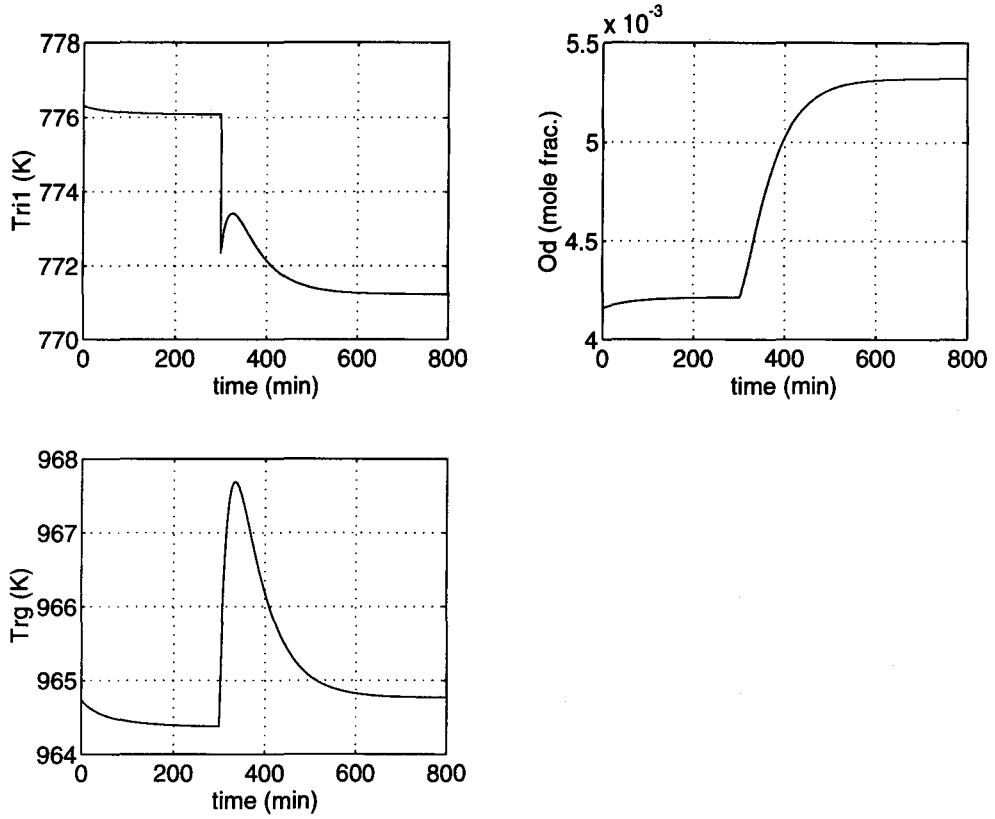


Figure 9: Step response on t = 300 min. F_a on working point value, F_s = 17040 kg/min.

In Figure 9 the effect of a decrease in F_s on O_d, T_{rg} and T_{ri1} is plotted. The stepresponse starts with an increase of the temperature T_{rg}, because less "cold" catalyst flows from the riser to the regenerator. This decrease of F_s means that less used catalyst is recirculated to the regenerator. This results in less coke to be burnt off. This leads again to a decrease of T_{rg} and an increase of O_d (see previous example). T_{ri1} shows a step like decrease, followed by an increase and then again a further decrease of T_{ri1}. In the model used for the FCC plant [Lee85], T_{ri1} depends time independently on the control variable F_s and time independently linearly on the output variable T_{rg}. These effects both take place at the same time, but the decreasing of F_s is first noticed, followed by the behavior of T_{rg}.

4.4. Steady state

The FCC plant reaches a new stable point after 200 min (if not unstable) after a step on an input is applied. If the steady state values at this point are analyzed, non-linearities appear.

4.4.1. Step response on F_a

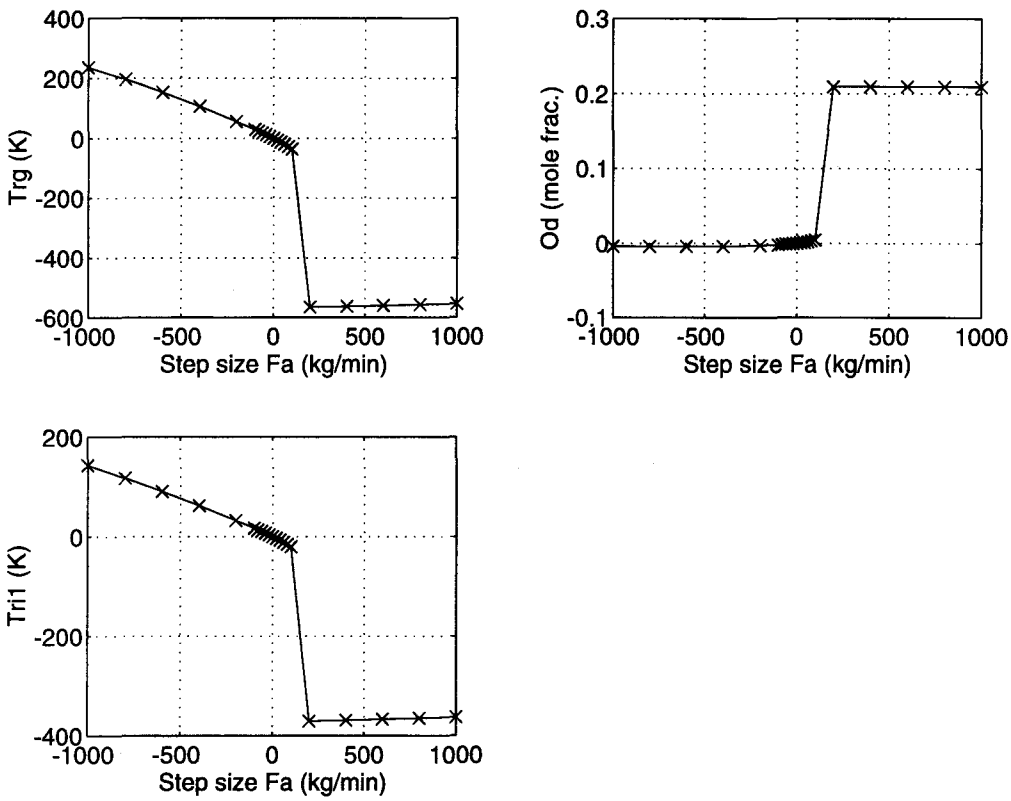


Figure 10: Scaled step response steady state values of F_a. × Are the measured values in the steady state situation.

Figure 10 shows the steady state values of the new equilibrium 200 minutes after the step input. The working point of F_a = 1521.1 kg/min is the reference value and is set to zero in Figure 10. The outputs are also set to zero on their working points. Thus when an input value of 1521.1 kg/min is used, the outputs are T_{ri1} = 776.2 K, O_d = 4.2e-03 and T_{rg} = 964.5 K and are set zero. For example F_a +100 kg/min results in an input to the FCC plant of 1621.1 kg/min. If the outputs of the FCC plant depend linearly on the inputs, a straight line is drawn between the points on the graphs. It is obvious that the outputs T_{rg} and T_{ri1} are linear in the range from -100 to 100 kg/min. Their corresponding correlation coefficients are $r(T_{rg})$=0.9985 and $r(T_{ri1})$=0.9984 (a straight line has a correlation coefficient of 1). O_d, however, has a correlation coefficient of $r(O_d)$=0.9669, which becomes even worse, when F_a is varied around the working point between the -1000 kg/min and 1000 kg/min. O_d approaches zero when the air inlet is decreased and increases to 0.21 (this is mole fraction of oxygen in air) when F_a is increased. Figure 11 depicts the time elapsed between the step and the saturation of O_d., in order to show that the larger the step on Fa is, the sooner the saturation level of 0.21 is reached.

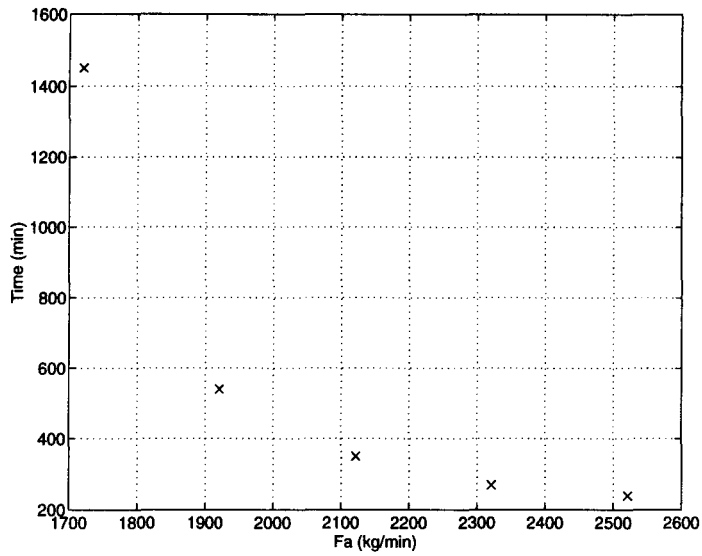


Figure 11: The time elapsed between start of the step F_a and saturation of O_d as a function of the absolute step value.

Simultaneously with the increase of the mole fraction O_d , the temperatures T_{rg} and T_{ril} drop, and approach the steady state situation, where no combustion takes place in the regenerator.

4.4.2. Step response on F_s

In Figure 12 is shown that O_d depends almost linearly on input step sizes of F_s (r=0.9998 for an input range of 200 kg/min and r=0.9950 for an input range of 2000 kg/min). This can not be concluded on T_{rg} and T_{ril} from which T_{rg} is the most non linear ($r(T_{rg})$=0.9757 against $r(T_{ril})$=0.9997 for an input range of 200 kg/min and $r(T_{rg})$=0.6565 against $r(T_{ril})$=0.9356 for an input range of 2000 kg/min).

As mentioned before if F_s makes a negative step, temperature T_{rg} augments after an initial increase and it ends at a higher temperature as the starting temperature. Likewise it can be told that if F_s makes a positive step but now T_{rg} increases more than the first decrease. This non linearity already starts when F_s increases +/- 50 kg/min.

Right at the beginning the step in temperature T_{ril}, due to an input step of F_s, is linear with the input F_s. The prove will be given in the next section, if the plant is compared to a linear model. The local extremes in time are linear with the stepsize too.

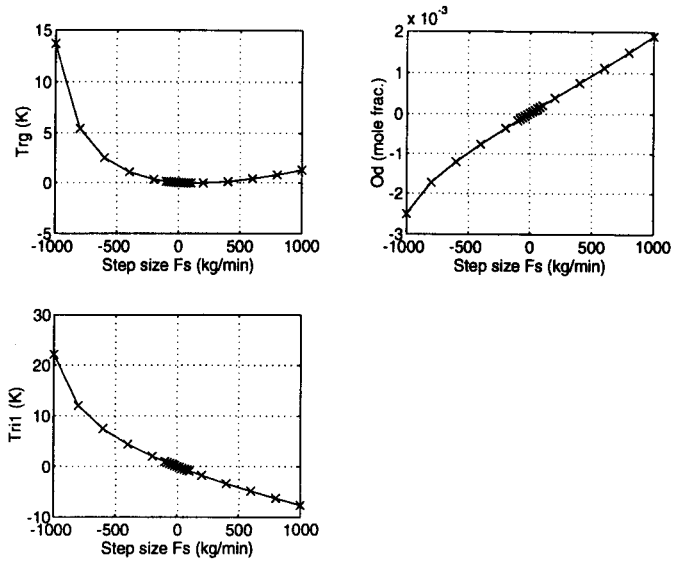


Figure 12: Scaled step response steady state values of F_s. × Are the measured values in the steady state situation.

4.4.3. Model versus plant

As described in chapter 2 the MPC controller uses a state space model to control the plant. In this section, the estimated model is compared to the non linear plant. This state space model is a linearization of the non linear FCC plant on its working point. The building of a state space model with a PBRNS signal and with the help of the PRIMAL module GUIDORZI did not lead to a model of the FCC plant as good as the analytically determined state space model. To compare the linear model with the FCC plant, step responses on F_a and F_s are compared

4.4.3.1. Step response on F_a

The model overestimates the static gain for T_{rg} and T_{ri1} by a little when a positive step is made (ranging from 22% to 5% for T_{rg} and from 34% to 5% for T_{ri1} when F_a ranges from 2521.1 kg/min to 1621.1 kg/min). O_d of the model becomes negative, even though this is physically impossible. It should be noticed that O_d is linearized around a working point. Therefore a negative value of O_d is possible. O_d of the plant approaches zero in the situation when O_d of the model is negative.

When a negative step is made, the outputs are underestimated. This can be ascribed to the fact that the plant is non linear. The plant reaches the state where no combustion takes place in the regenerator while in the model still combustion is modeled because of linearity. Therefore O_d is modeled so badly. In the worst case there is a factor 5.7 overestimation at steady state and F_a = 2521.1 kg/min. Both T_{rg} and T_{ri1} have steady state values of 400 K and O_d has a steady state value of 0.21.

For the control problem this means, that O_d is difficult to control with F_a.

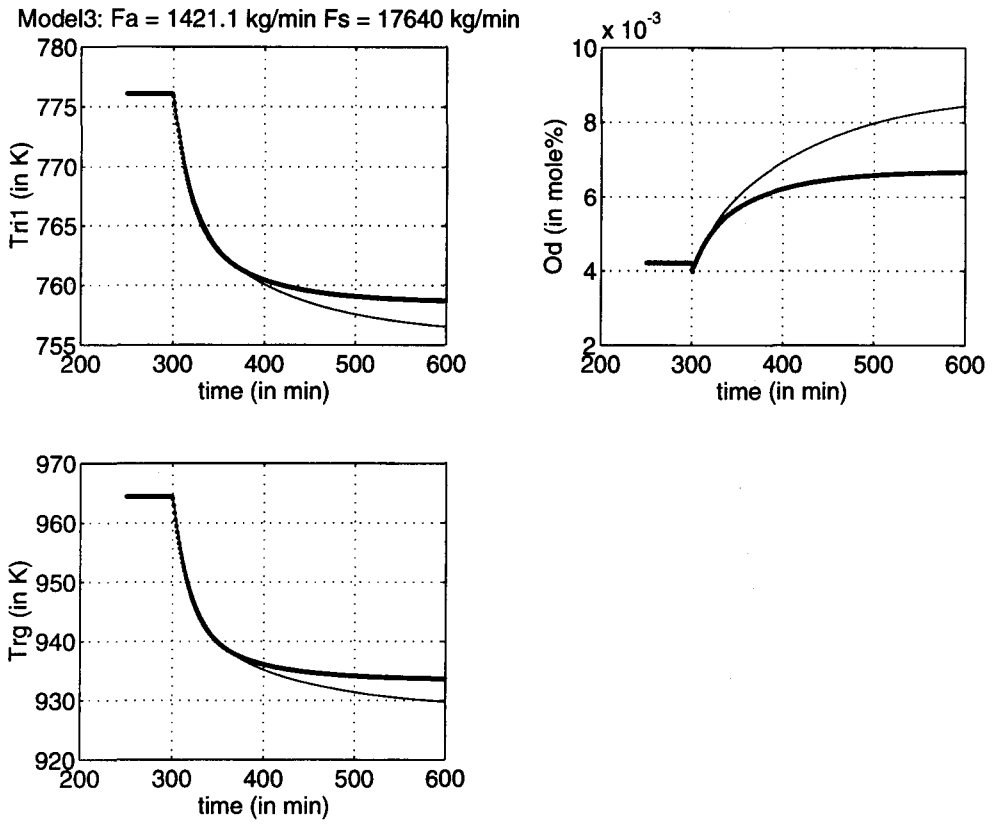


Figure 13: Plant (thin line) and Model(thick line). Start step at t=300 min. F_a goes from 1521.1 kg/min to 1421.1 kg/min. F_s is on working point.

4.4.3.2. Step response on F_s

As mentioned in section (4.2.2), O_d is more linear than the two temperature outputs. The maximum deviation of O_d between the plant and model is 11.9% for an input range of F_s of 2000 kg/min. When F_s does not deviate too much from the working point (100 kg/min), plant and model match well. In Figure 14 the match of the model to the plant is shown. If F_s deviates more than 100 kg/min from the working point, only the step made by the output T_{ri1} is the same for model and plant. The time point as well as the values of the extremes of T_{ri1} are equal for model and plant too. The dynamics of T_{rg} of the linear model is the same as the real plant in the first 40 min. After 40 min the model underestimates the values of the real plant, but no more than 2%.

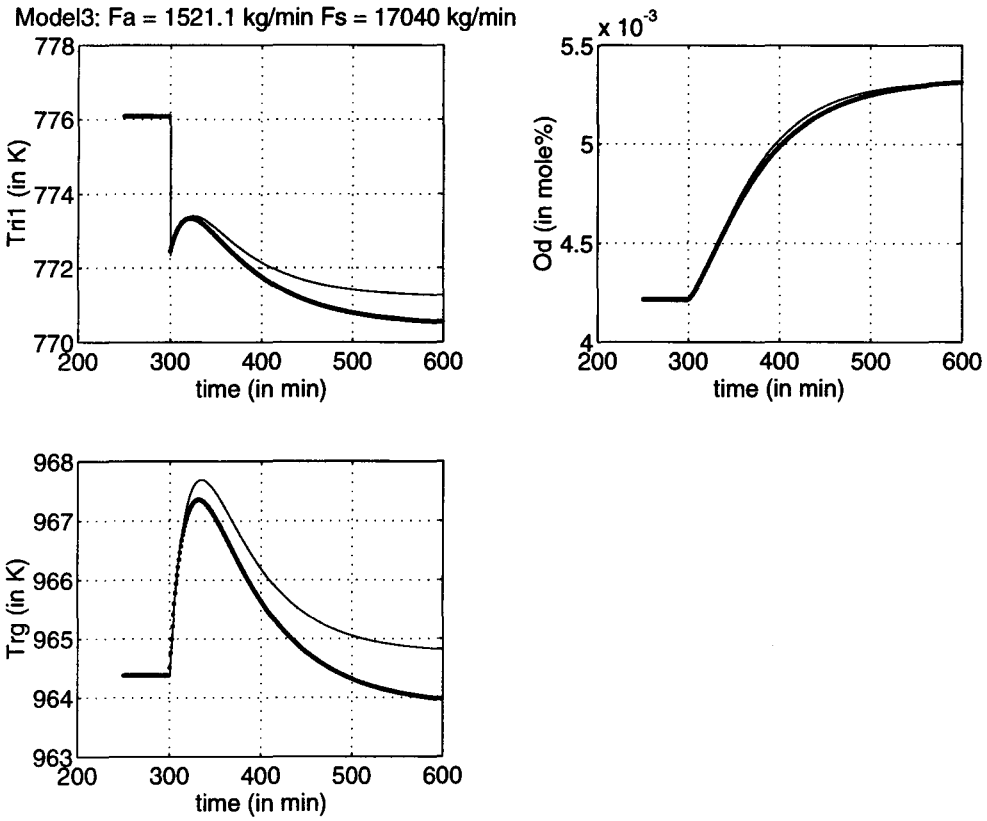


Figure 14: Plant: Thin line, Model: Thick line. Step starts at t=300 min. F_a is on working point and F_s goes from 17640 kg/min to 17040 kg/min.

5 . Simulations

5.1 . Introduction

This chapter contains the results of the analysis of the MPC controller. The MPC controller is tested on a few typical control problems and is compared with a Linear Quadratic Gaussian controller (LQG controller). The MPC controller has to control the non-linear model of the FCC plant, described in chapter 4. Furthermore some properties of the Logical Controller are shown and some rules of thumb are given to tune the MPC controller.

The start situation is the same as for all the simulations. In Table 1 the values of the inputs and outputs are given of this steady state start situation:

output: T_{ril}	776.2 K
output: O_d	4.2e-03 mole frac.
output: T_{rg}	964.5 K
input: F_a	1521.1 kg/min
input: F_s	17640 kg/min
disturbance: F_{oil}	2438 kg/min

Table 1: working point values of the inputs and outputs of the FCC plant.

5.2 . LQG controller

The MPC controller is compared to a LQG controller to learn more about the performance of the controller. The MPC controller can be compared to the LQG controller under certain circumstances [Lee94].

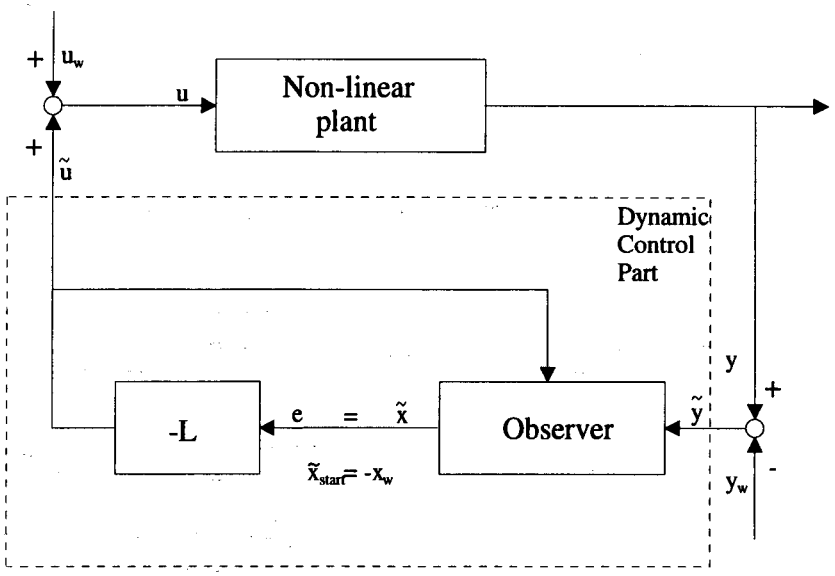
The software program Matlab™ 4.0 has been used to build a discrete LQG controller. This LQG control loop is depicted in Figure 16. In order to design a LQG controller which controlled the outputs y, the control strategy has been split in a static control and a dynamic control part. By doing so the error vector e equals $\underline{0}$, when the new working point has been reached. That means $\tilde{u}$ equals $\underline{0}$ and only the static control signal u_w is needed to keep the outputs of the plant at the wanted setpoint values. In the Control System Toolbox [Mat90] only LQG controllers can be designed which use a function of state variables as input to the regulator.

To determine u_w and x_w , the state space model is used. The LQG controller uses two control variables, thus two output setpoints must be set. Otherwise the problem will become infeasible. Assumed:

$$\begin{aligned} x[n+1] &= \Phi x[n] + \Gamma_u u[n] \\ y[n] &= Cx[n] + D_u u[n] \end{aligned} \tag{5.1}$$

Because at steady state $x[n+1]=x[n]$, the setpoints u_w and x_w can be calculated off line straightforward.

$$\begin{aligned} u_w &= \{C(I-\Phi)^{-1}\Gamma_u + D_u\}^{-1} \cdot y_w \\ x_w &= \{I-\Phi)^{-1}\Gamma_u \cdot u_w \end{aligned} \tag{5.2}$$



u: control vector
y: output vector
y_w: output setpoint vector
u_w: control vector at setpoint
x_w: state vector at setpoint
e: error vector
$\tilde{y}$: dynamic output vector
$\tilde{u}$: dynamic control vector
$\tilde{x}$: dynamic state vector
-L: regulator

Figure 16: Scheme of LQG controller and plant, with the splitting in a static and dynamic part.

If the objective function of MPC in equation (2.24) is compared to the objective function of the LQG controller in equation (5.3) [Kok90], then the objective function of the MPC controller resembles the objective function of a LQG controller the most when the same length is chosen for both the prediction and control horizon and P_e is chosen a zero matrix in the objective function of the LQG controller.

$$J = \sum_{i=i_0}^{i_e-1} \{ y_i^T \Gamma_y^T \Gamma_y y_i + u_i^T \Gamma_u^T \Gamma_u u_i \} + y_{i_e}^T P_e y_{i_e} = \sum_{i=i_0}^{i_e-1} y_i^T \Gamma_y^T \Gamma_y y_i + \sum_{i=i_o}^{i_e-1} u_i^T \Gamma_u^T \Gamma_u u_i + y_{i_e}^T P_e y_{i_e} \qquad (5.3)$$

where start values y_{i_0} equal $\underline{0}$, y_{i_e} are the end state values, Γ_y, Γ_u and P_e are weight matrices. $\Gamma_y^T \Gamma_y$ has to be semi positive definite and $\Gamma_u^T \Gamma_u$ has to be positive definite in order to find the optimal controller used in the Matlab™ routines.

The length of the horizons must be equal to the settling time of the process to be controlled (see below). The stepresponses displayed in chapter 4 show a settling time of 200 min.

Both the MPC controller as well as the LQG controller use a Kalman filter to filter the measured outputs of the plant. Both controllers used the same parameter settings for the Kalman filter. The MPC controller is compared with the LQG controller to simulate on setpoint tracking using the non-linear FCC model. The setpoints are T_{ril} = 790 K and T_{rg} = 980 K.

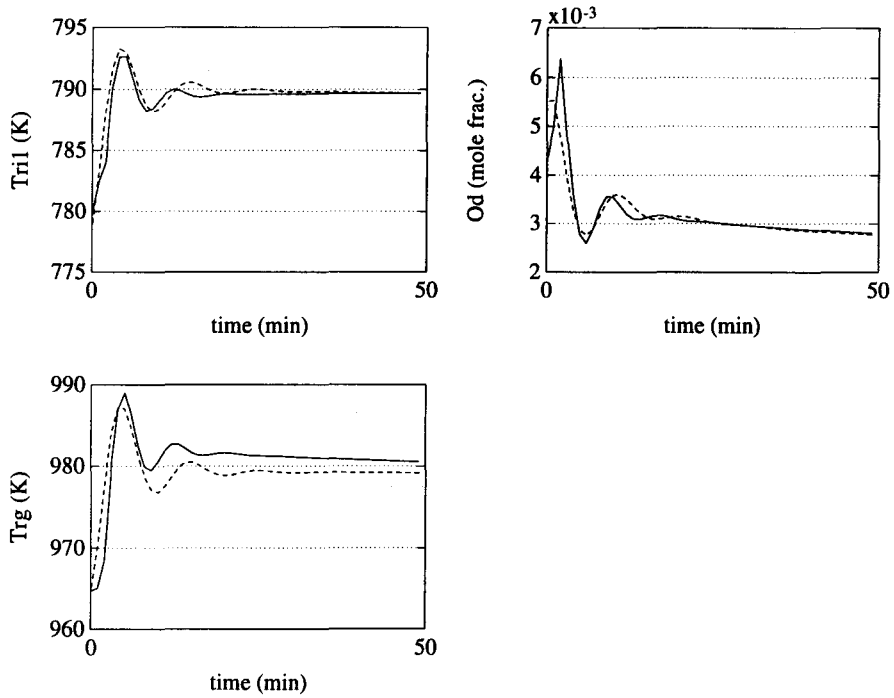


Figure 17: The outputs of the FCC plant. The MPC controller is the dashed line. The LQG controller is a solid line. Setpoints: T_{ri1}=790 K and T_{rg}=980 K. Both m and p are 200.

Figure 17 shows that both the outputs start with a little overshoot. The overshoot is approximately the same for both the controllers. For T_{ri1} the MPC controller has a maximum overshoot is 3.3 K and for LQG for 2.6 K. For T_{rg} the MPC controller has a maximum overshoot is 7.1 K and for LQG for 9.0 K. The oscillatory behavior stops in both cases after 29 minutes. In the steady state situation after 100 min the steady state values of MPC controller show a small offset (T_{rg}(offset)=0.5 K and T_{ri1}(offset)=1.0 K), while the steady state values of the LQG controller show a negligible offset.

Although the outputs of the plant show great resemblance, the input variables do not, as shown in Figure 18.

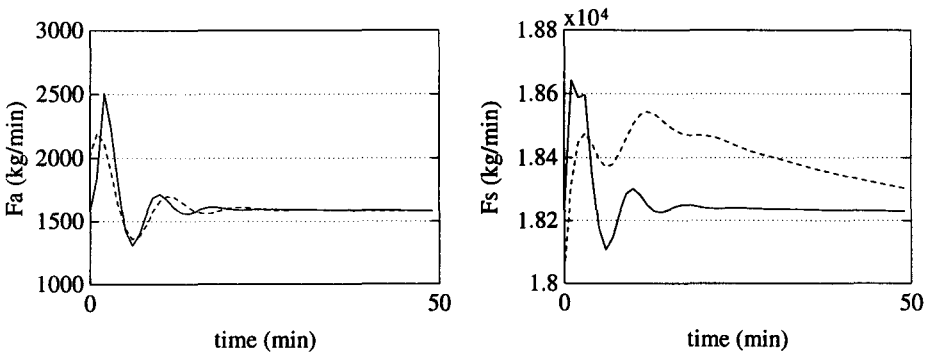


Figure 18: The inputs to the FCC plant. The MPC controller is the dashed line. The LQG controller is a solid line. Both m and p of the MPC controller are 200.

In Figure 18 the most striking difference between the two controllers is the behavior of the control variable F_s. In case of the LQG controller F_s has an oscillatory behavior, while in the case of the MPC controller F_s begins with a sharp increasing of F_s and then slowly decreases to a 1.82e+04 kg/min.

The difference in the behavior of F_s is a result of the different objective function for the MPC controller and the LQG controller. In the objective function of the MPC controller $\Delta u(k)$ is optimized, where $\Delta u(k)$ is the deviation from current value $u(k\text{-}1)$. In the objective function of the LQG controller $u(k)$ is optimized, where u(k) is the deviation from the value of the manipulated variables at the working point u_w. Therefore the objective of LQG controller is to return to the working point of the manipulated variables u_w as fast as possible and the objective of the MPC controller is to deviate as little as possible from the last invoked value to the process.

The reason why the outputs of the plant still have the same behavior results from control variable F_a having much more impact on the output variables of the FCC plant than F_s (see Figures 5 and 6). The manipulated variable F_a shows a similar behavior in both cases. The differences are the first overshoot peak of F_a(LQG) is about 600 kg/min higher than the overshoot peak of F_a(MPC) and the F_a(MPC) oscillates a little slower, but otherwise there are no real differences.

5.3. Constraint handling

One of the major advantages of a MPC controller over most of the other controllers is the straightforward way it handles constraints. In Figure 19 and Figure 20 a typical example is given of a constraint on one of the inputs. The constraint is given by the user. These figures show the behavior of the inputs and outputs of the plant for three values for the upper constraints set on the manipulated variable F_a.

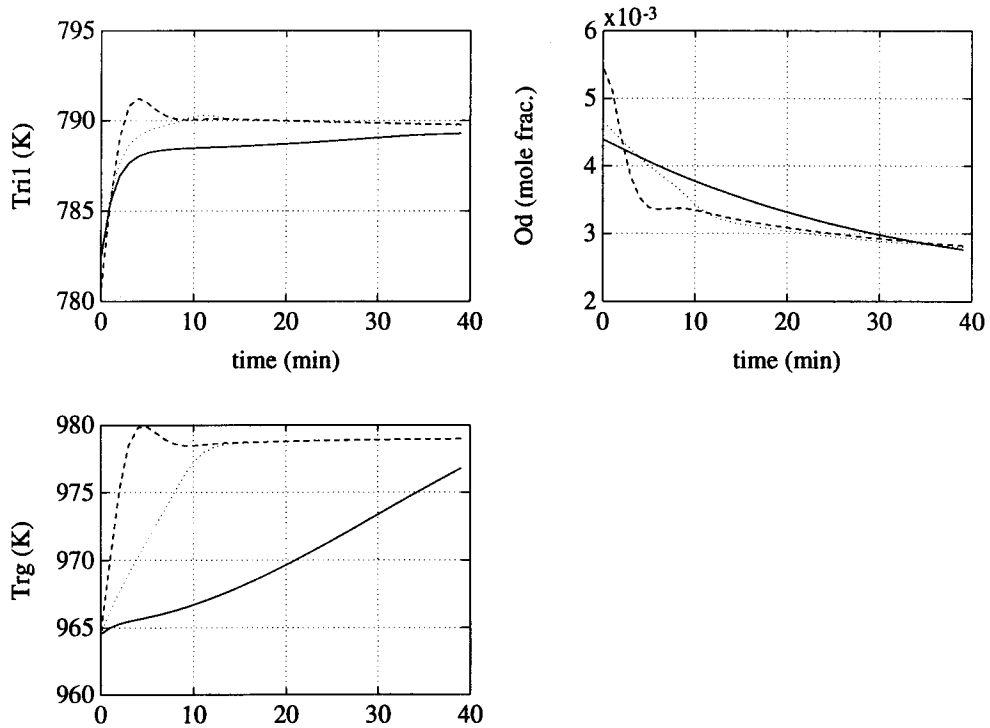


Figure 19: The outputs of the FCC plant unconstrained (dashed line), upper constraint on F_a=1700 kg/min (dotted line), upper constraint on F_a=1600 kg/min (solid line).

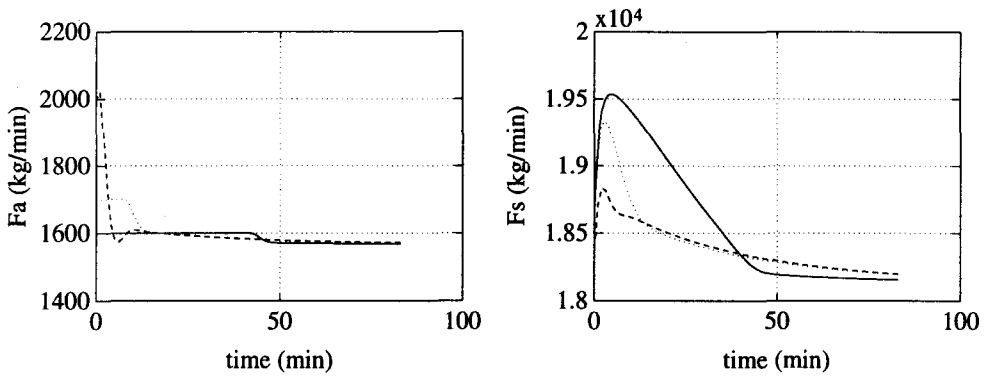


Figure 20: The inputs of the FCC plant unconstrained (dashed line), upper constraint on F_a=1700 kg/min (dotted line), upper constraint on F_a=1600 kg/min (solid line).

In Figure 19 the first 40 minutes of the simulation are shown to get a good image of the output behavior for the different constraint values before reaching the setpoint. The solid line reaches its setpoints after 70 minutes. As expected the simulation with the upper constraint set on infinite (unconstrained case) reaches the setpoints value T_{ri1}(setp)=790 K and T_{ri1}(setp)=980 K the fastest, followed by the simulation of the upper constraint set on F_a(upper)=1700 kg/min. When the constraint is set on F_a(upper)=1600 kg/min, it takes about 42 minutes before the controller leaves the constrained value. In Figure 20 the controller compensates for the constraint on F_a, through the increase of the recirculation rate of the catalyst F_s

5.4 . Constraint release

An output often has to stay between an upper and a lower constraint, but in for example in a start situation it is not always possible to find a solution with the output between the two constraints. A temporarily release of the constraints could solve the problem of not finding a solution. Therefore in the Logical Controller, a constraint releasing algorithm has been built, which regulates the release of constraints.

In Figure 21 and Figure 22 two simulations of exactly the same problem are shown. In the first case the constraints are not allowed to be released, while in the second case they are allowed to be released.

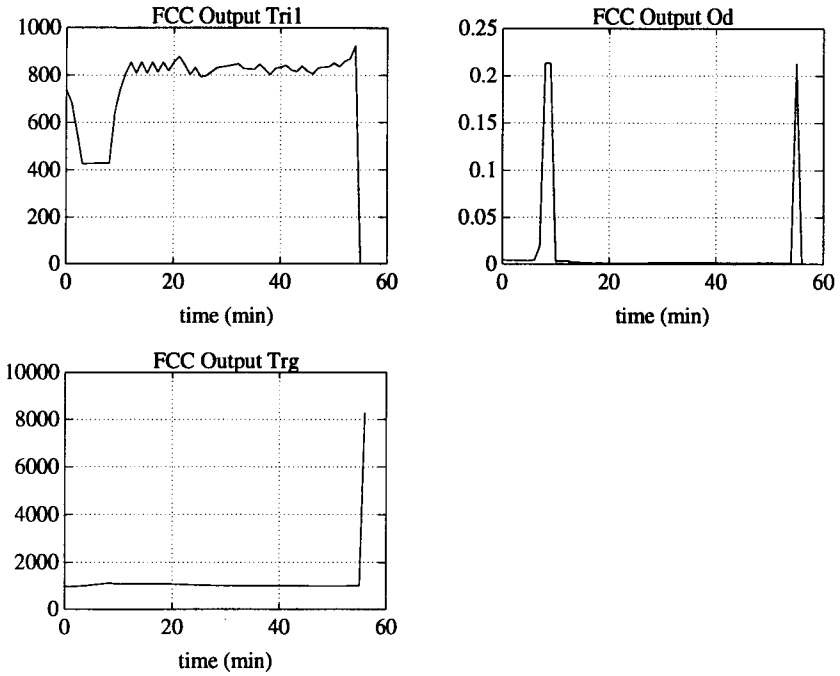


Figure 21: The outputs with 2 setpoints, T_{ri1}=780 K and T_{rg}=985 K. Constraints on T_{rg}, 980 K< T_{rg} < 990 K. The constraints can not be released. The plant becomes unstable after 58 minutes.

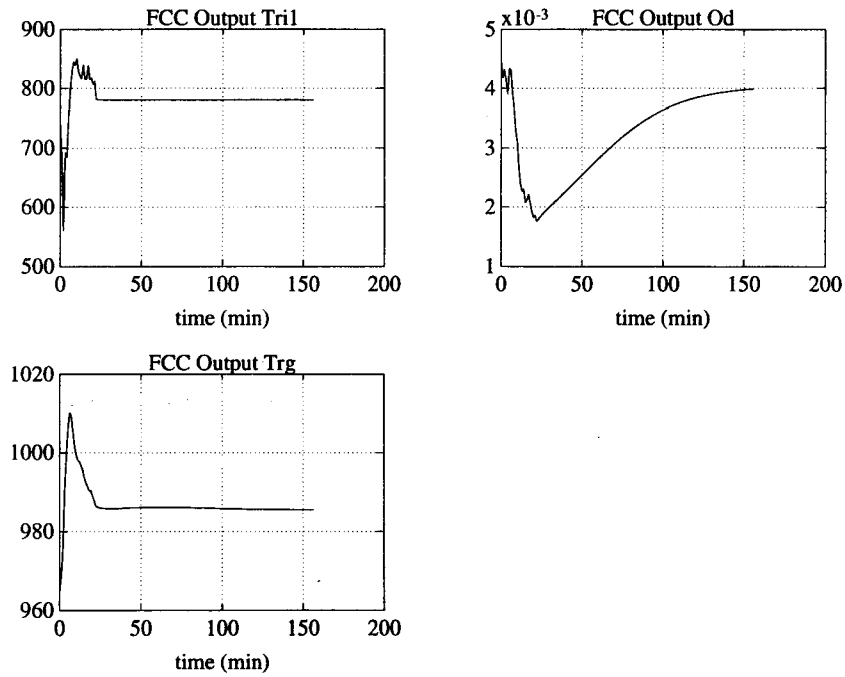


Figure 22: The outputs with 2 setpoints, T_{ri1}=780 K and T_{rg}=985 K. Constraints on T_{rg}, 980 K< T_{rg} < 990 K. The constraints are allowed to be released.

In the second simulation the constraints on T_{rg} are released in the first 20 minutes. Every sample the optimization algorithm of the MPC controller tries to find a feasible solution of the control problem with regards to the constraints. The optimization routine always finds a set of values for the manipulated variables. If the solution is not feasible, the values found by the optimization algorithm are still the best values to be implemented. The non feasible solution found is the solution the most closely to the constraints, although violated. Through the release of constraints the found solution can become feasible and the values of the manipulated variables can be implemented. The peaks on the output variable T_{rg} are the result of the effort of the optimization algorithm to find a feasible solution with respect to the constraints.

The plant outputs reach their setpoint values after 20 minutes. In the first simulation the constraints on T_{rg} could not be released and therefore no feasible solution could be found for which the output T_{rg} was between the constraints. The plant becomes unstable after 58 minutes of simulation.

5.5 . Disturbance prediction

Another major advantage of the MPC controller is that it can take the behavior of disturbances in the future into account. The controller gets the information of future disturbance changes from the user or another source. The MPC controller is provided with this info when the future state variables are calculated (equation 2.17).

$$x(k+i) = \Phi^i \hat{x}(k) + \sum_{i=1}^{p} \Phi^{i-1}(\Gamma_u(k-1) + \Gamma_d d(k) + \Gamma_z \hat{z}(k)) + \sum_{i=1}^{s} \Phi^{i-1}\Gamma_d \Delta d(k+i) + \sum_{i=1}^{m} \Phi^{i-1}\Gamma_u \Delta u(k+i) + \sum_{i=m+1}^{p} \Phi^{i-1}\Gamma_u \Delta u(k-m+i), \quad for\ i > m \tag{5.4}$$

where $\Delta d(k+i)$ is the difference between the measured value and the predicted value at t_{k+i}. Because the measured value of d is chosen to be kept constant over the complete prediction horizon, the measured value can be subtracted from the predicted value.

Figure 23 and Figure 24 show the plant in - and outputs as a result of disturbance in the oil feed to the riser F_{oil} at t = 40 min when the disturbance is predicted and when the disturbance is only measured. In both the cases the MPC controller eliminates the disturbance and the outputs go back to their setpoints of T_{rg}(setp)=964.5 K and T_{ril}(setp)=776.2 K, respectively.

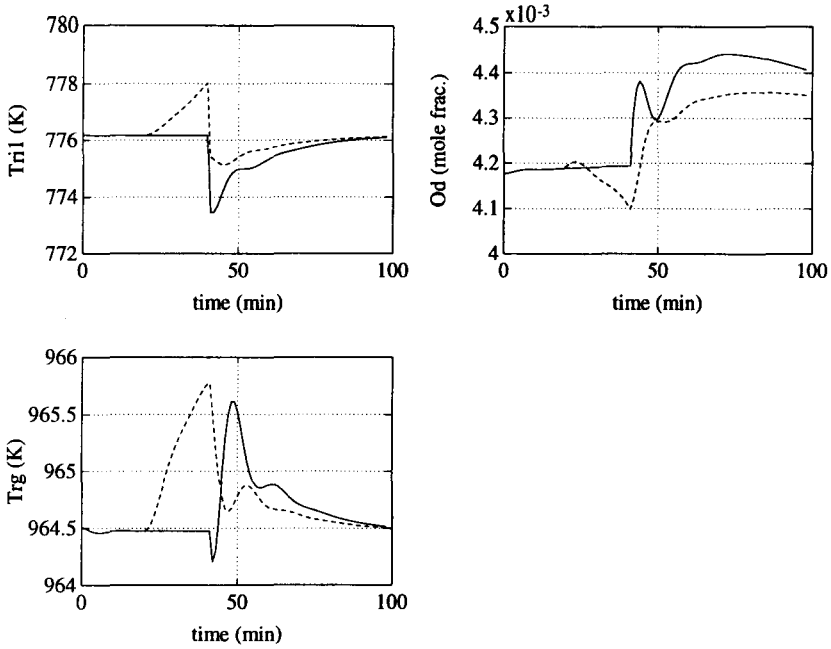


Figure 23: The outputs of the FCC plant. The dashed lines are the outputs when the disturbance is predicted. The solid lines when the disturbances are not predicted.

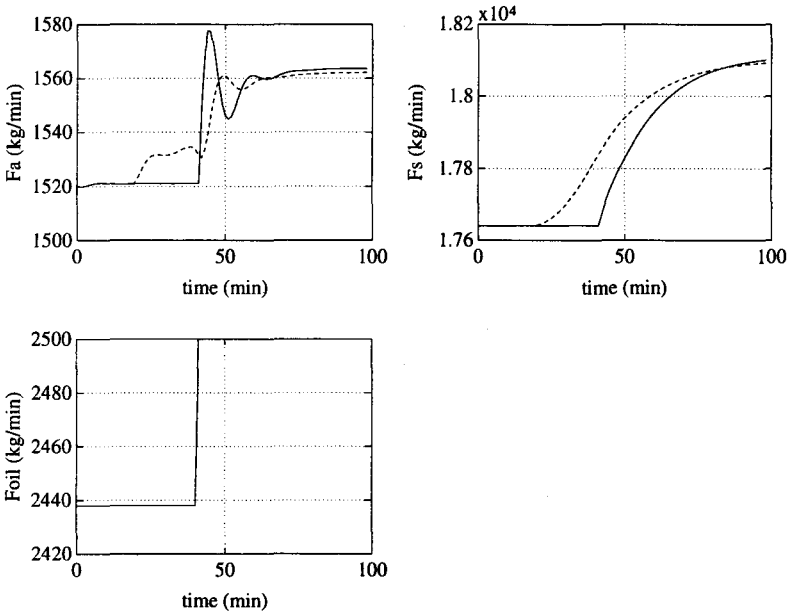


Figure 24: The inputs of the FCC plant. The dashed lines are the outputs when the disturbance is predicted. The solid lines when the disturbances are not predicted. The disturbance input is F_{oil} and it changes at t=40 min.

Because the prediction horizon lasts 20 minutes the controller starts taking action 20 minutes before the disturbance is predicted. The controller takes as fast as possible measures to eliminate the disturbance change at t = 40 min. This is done by an increase of the manipulated variables F_a and F_s. As a result of the increase, the output variables increase too. 8 Minutes after t = 40 min, the output variable T_{ri1} is within 0.6 K of the setpoint. In the unpredicted case it takes 11 minutes more for T_{ri1} to return into a 0.6 K band. In the predicted case the disturbance is corrected 2.3 times as fast as in the unpredicted case.

The comparison of the two different situations with respect to the output variable T_{rg} is more difficult. The maximum deviation from the setpoint is 0.5 K higher in the predicted case as in the unpredicted case. Both outputs return to their setpoint at the same time, but the deviation from the setpoint decreases faster in time in the predicted case than in the unpredicted case.

The MPC controller prepares the process on the coming disturbance, but the slow dynamics of T_{rg} (in comparison to those of T_{ril}) prevent the output T_{rg} from returning to the setpoint as fast as the output T_{ril}. Therefore the correction of output T_{rg} is slower than the correction of the output T_{ril}.

It can be concluded that when a disturbance is predicted, the MPC controller rejects the disturbances faster in comparison with an unpredicted disturbance.

5.6 . Prediction horizon

The user must tune the prediction horizon between a lower limit and a upper limit. The upper limit is the settling time of the process. The reason for a lower limit will be explained by showing a simulation of the control of an inverse response in Figure 25.

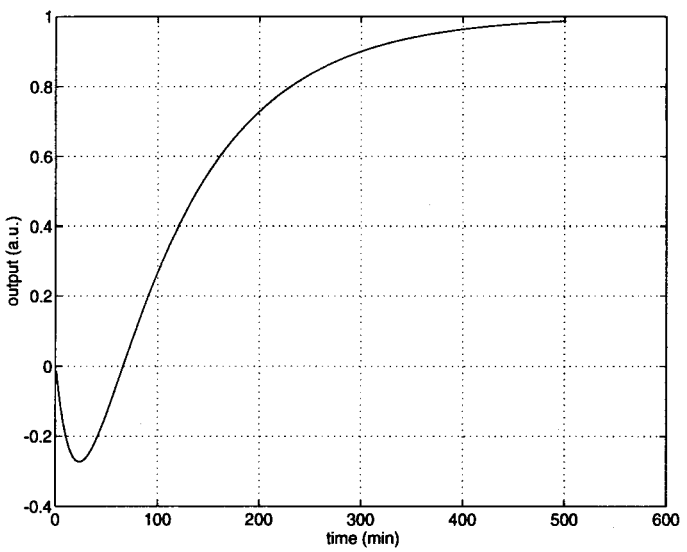


Figure 25: Stepresponse of an inverse response model.

As mentioned before the prediction horizon is defined as the time over which the MPC controller predicts the output and state variables. In next 4 figures is shown the effect of a prediction horizon chosen too short and chosen long enough.

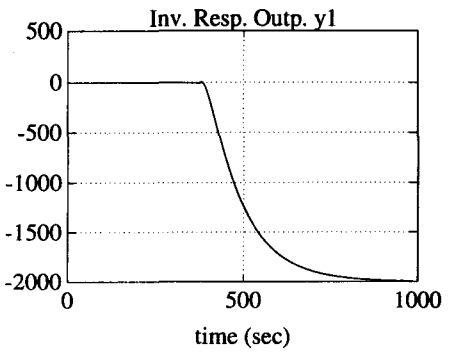


Figure 26: The outputs of the process with the prediction horizon p=20.

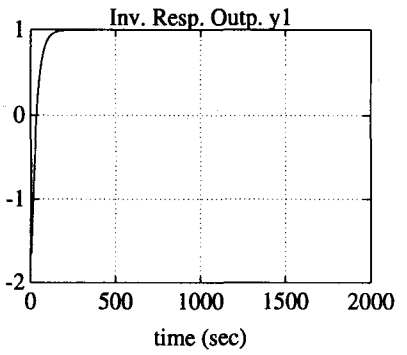


Figure 27: The outputs of the process with the prediction horizon p=300.

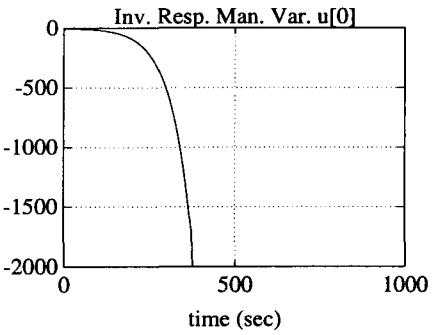


Figure 28: The inputs of the process with the prediction horizon p=20.

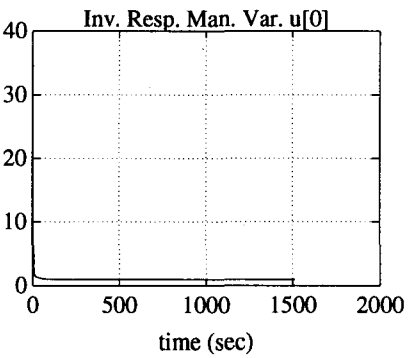


Figure 29: The inputs of the process with the prediction horizon p=300.

In the first simulation (Figure 26 and Figure 28) the controller has no information on the inverse response, because the prediction horizon is chosen 20 min. The calculated manipulated variables are not correct, because of the lack of information the controller has.

In the second simulation (Figure 27 and Figure 29) the prediction horizon is chosen long enough in order to obtain a stable output to see that the stepresponse is an inverse response. As a consequence of the inverse response, first the output variable must become negative in order to obtain a stable output variable at setpoint 1. Looking at the output variable, first the output has a value of -1.7 (a deviation of 2.7 of the setpoint) but in the continuation the output reaches the setpoint and does not deviate from this setpoint anymore.

As a rule of thumb the minimal length of the prediction horizon is when the plant outputs reach steady state values which are comparable with the values of the model outputs at the endpoint of the prediction horizon.

5.7. Control horizon

The control horizon is like the prediction horizon a tuning parameter. But the effect of the change in the control horizon is stronger and the choice of the prediction horizon is bound to a maximum value. A control horizon longer than the prediction horizon would lead to a situation that the control moves are calculated in a region where no information is available. When the control horizon is increased the controller acts faster both on disturbance reduction as well as setpoint tracking, the controller behaves more "aggressive" [Mor91]. The drawback of the more aggressive behavior is the fact that the control loop is less robust to model plant mismatch.

When the control horizon is increased, the number control moves of the of manipulated variables to optimize is extended. These extra variables function as extra degrees of freedom. The optimization routine of the MPC controller uses these extra degrees of freedom to act faster on disturbances and on setpoint changes. If the control horizon is a long (± 30 control moves) the model plant mismatch is assumed to be small. So as a rule of thumb: the control horizon has an upper limit that depends on the size of the model plant mismatch. The smaller the model plant mismatch the longer the control horizon can be chosen.

Besides the control horizon parameter there is another tuning parameter related to manipulated variables. This is called input blocking. The manipulated variables are not allowed to move at every sample of the control horizon. An input block of 4 means that a block of 4 succeeding control moves is optimized. Therefore they all have the same value. Input blocks make the controller less aggressive.

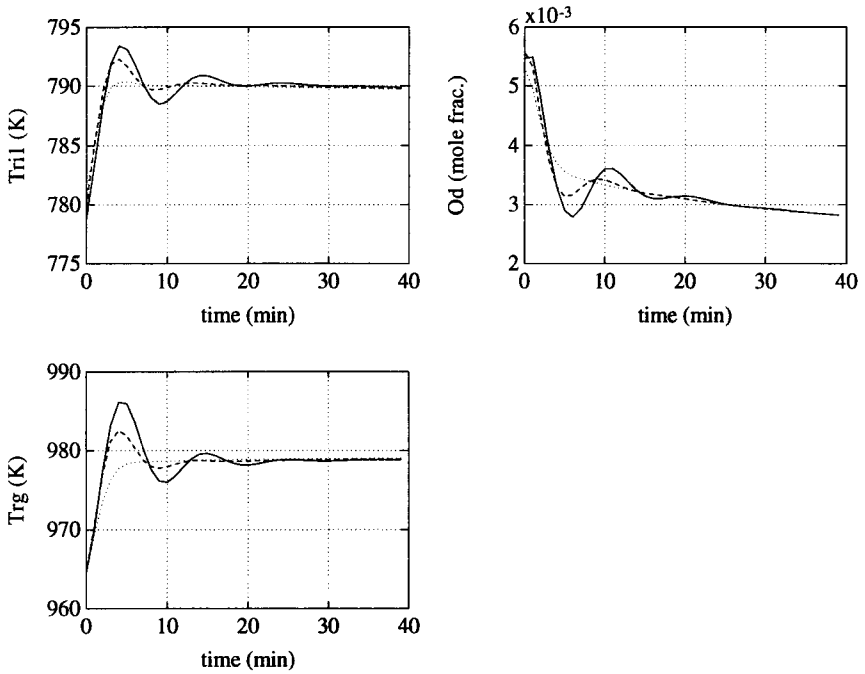


Figure 30: The outputs of the FCC controller with 2 setpoints: T_{ri1}=790 K and T_{rg}=980 K. m = 20 in all cases. 20 x 1 sample input block (solid line), 10 x 2 sample input blocks (dashed line), 5 x 4 sample input blocks (dotted line).

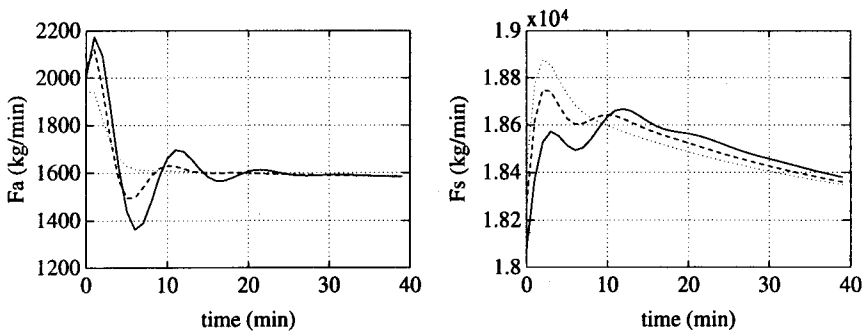


Figure 31: The inputs of the FCC controller with 2 setpoints: Tri1=790 K and Trg=980 K. m = 20 in all cases. 20 x 1 sample input block (solid line), 10 x 2 sample input blocks (dashed line), 5 x 4 sample input blocks (dotted line).

Figure 30 and Figure 31 show the simulations for the setpoints $T_{ril} = 790$ K and $T_{rg} = 980$ K. The control horizon is m=20 and only the number of input blocks differs. It is obvious that the MPC controller with the longest input block shows the least aggressive behavior.

5.8. Offset

The settings of the Kalman filter determine the performance of the controller for a great deal [Bro83]. As mentioned in chapter 2 the Kalman filter expresses the ratio of model noise and measurement noise and is used to improve the prior estimation of the model state variables. To learn more about the performance of MPC controller when different Kalman filters settings are used, a number of simulations is done. Only the measurement standard deviations are changed. In Table 2 relevant parameters can be found. Figure 32 shows a typically response of the output variables and Figure 33 expresses the values of the outputs at $t = 300$ min.

	T_{ril} (K)	O_d (mole frac.)	T_{rg} (K)
setpoint	780	-	955
std. dev. model noise v	3.16e+00	2.00e-03	1.00e+00
std. dev. meas. noise w	1.00e-10 - 1.0e+00	2.00e-12 -2.0e-02	1.00e-10 - 1.0e+00

Table 2: Relevant parameters of the series of simulations

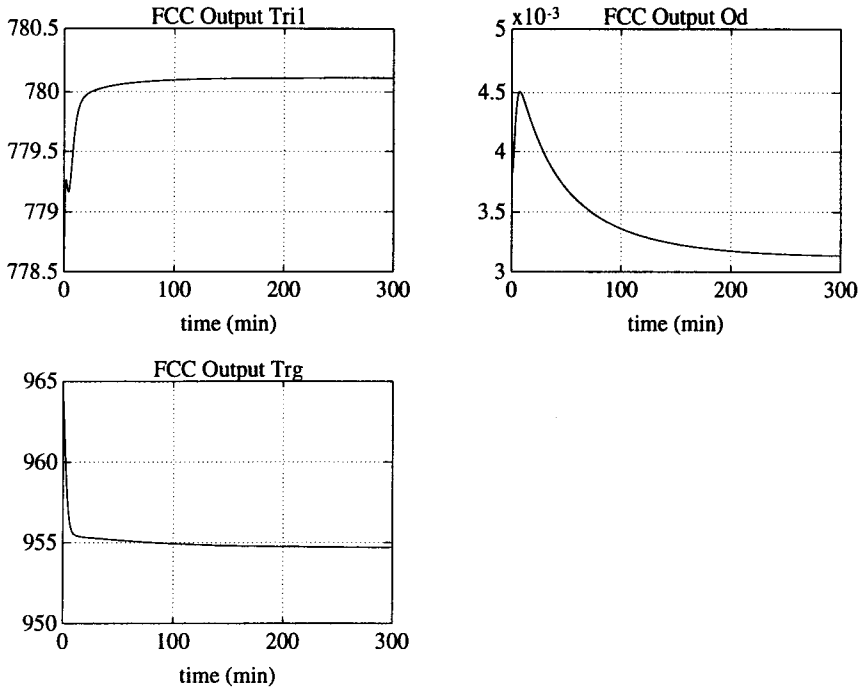


Figure 32: Typical response of the outputs when setpoint T_{ri1}=780 K and T_{rg}=955 K. p=200, m=60 and 15 x 4 sample input blocks.

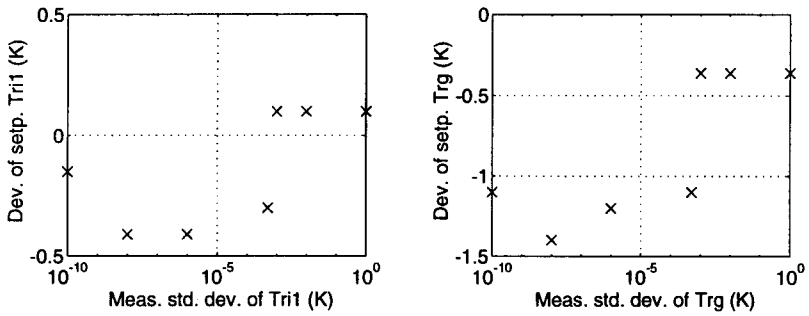


Figure 33: The deviations of the setpoints for T_{ri1} and T_{rg}.

Figure 33 shows that generally the setpoints offsets increase while the estimated measured standard deviations decrease. A possible reason for this is that the way the future state variables are predicted. With the updated state variables the next state variables are calculated. So only the first sample of the prediction horizon is influenced directly by plant measurements. The influence of the measurements decrease every next sample of the prediction horizon.

If the relationship between the inputs and the outputs of the plant is unambiguous (as is in this case), a combination of output values at steady state correspond to a combination of manipulated variables values. At steady state the effect of measurements is negligible, because in the equilibrium situation a small deviation from the state variables values does not lead to a real different plant behavior. If there is a model plant mismatch in the static situation, the calculated manipulated variables will lead to other output values as wanted; the offset.

When not at steady state the updated state variables are of influence to the dynamics of the plant. Therefore the calculated manipulated variables values are different for different updated state variables. Thus for different settings of the Kalman filter. Because of non linearities of the plant, coincidentally the simulations with the best updated state variables, give the worst performance.

6 . Conclusions and recommendations

6.1 . Conclusions

6.1.1. Model Predictive Control

One of the objectives of this graduation project was the analysis of the behavior of the MPC controller.

The prediction horizon and the control horizon are two parameters to tune the MPC controller. Some rules of thumb were found for the choice of the prediction and the control horizon in order to get a stable control loop. The maximum length of a prediction horizon is the length for which the model used to control the plant reaches the steady state situation. The minimal length of the prediction horizon is when the plant outputs reach steady state values which are comparable with the values of the model outputs at the endpoint of the prediction horizon. The prediction horizon should be chosen between the minimal and maximal prediction horizon and should be as long as the model does not deviate too much from the plant behavior.

After the prediction horizon has been chosen, the control horizon should be chosen. The number of control moves can be used to determine the "aggressiveness" of the controller. The best is to start with one control move and in order to increase aggressiveness to increase the number of control moves until a wanted control behavior is reached. The better the model describes the plant the more aggressive the controller can be.

If disturbance values can be predicted, this is seen as a measurement in the future by the MPC controller. The controller starts to take the predicted disturbance values into account as soon as it is within the prediction horizon and the calculated manipulated variables are adjusted for the predicted values. As a consequence, when a change in the disturbance values can be predicted, this does not necessarily mean that the average deviation of the setpoints is less than in the unpredicted case, when looking at a period ranging from the start of the anticipation of the controller in the predicted case to the complete correction in the unpredicted case.

This MPC controller has a so-called Logical Controller. One of the tasks of the Logical Controller is to release constraints temporarily if necessary and if possible. The temporarily release increases the robustness of the MPC controller as a whole. This function of the controller works well.

Most of the setpoints show offset. This is due to the fact this control algorithm handles the information from plant output measurements in a way only the first sample of state variables of the prediction horizon is updated with measurement information. The other state variable are updated via the simulation over the prediction horizon.

Because the MPC software is thoroughly tested in many different simulations and corrected where necessary, the reliability of the software has been improved.

6.1.2. Software

One of the objectives of this graduation project was the implementing of a MPC controller in the software tool PRIMACS. The development of this tool is still in progress. The MPC controller, described in this graduation project, was one of the first controllers build into this tool.

The plant, the controller and the program which validates and implements input parameters to the MPC controller were decoupled into independent structures. Each of the structures can be changed without affecting the other two.

The user can manipulate the input parameters in time because of the input and validation method for the input parameters of the MPC controller on basis of the scenario approach. The user will use questions pages to enter new input parameters.

The Logical Control unit increases the robustness of the MPC controller (e.g. constraint release), but decreases the reliability of the MPC controller. The added Logic Control unit has a decision strategy, which increases the complexity of the program. With this complexity the chance on errors increases, of which the consequences are not always visible.

6.2. Recommendations

6.2.1. Model Predictive Control

With the linear model of the non-linear FCC plant the MPC controller sometimes had great difficulties of controlling O_d, because of the strong non linearities in the behavior of O_d. A non-linear MPC controller can perhaps control O_d and additionally diminish the offset from the setpoints. The latter because of a smaller model plant mismatch compared to the linear MPC controller.

Further investigation on the offset problem is needed. Perhaps the use of a disturbance observers will solve most of the offset problems, but also an other way of using the measurement data to correct for model plant mismatch is needed where not only the first set of state variables is updated, but over state variables over the complete prediction horizon.

The controller always tries to stay within the constraint bands. In the simulations no measurement noise was added. It would be interested to see the behavior of the MPC controller when measurement noise is added. Because when measurement noise is added the measurement value of a constrained output could violate a constraint, although the real output value does not violate the constraint.

6.2.2. Software

The introduction of a Scenario Manager in PRIMACS. This Scenario Manager can be seen as an interface between the user and the simulation. The Scenario Manager guides the user through the manipulation of the process and the controller, both on line as well as off line. The Scenario Manager manages the data the user wants to have, like plots of the outputs of plant.

7 . References

[Bal90] : J.G. Balchen, D. Ljunquist, S.Strand, "State space predictive control", *Report no 90-90-W, Division of engineering cybernetics, NTH, Trondheim*

[Bro83] : R.G. Brown, "Introduction to random signal analysis and Kalman filtering", *John Wiley & Sons, 1983*

[Eat92] : J.W. Eaton, J.B. Rawlings, "Model predictive control of chemical processes", *Chemical engineering science, Vol. 47, No. 4, pp. 705-720, 1992*

[Gar89] : C.E. García, D.M. Prett, M.Morari, "Model Predictive Control: theory and practice — a survey", *Automatica, Vol. 25, No. 3, pp. 335-348, 1989*

[Gil81] : E. Gill, W. Murray, M.H. Wright, "Practical optimization", *Academic Press, 1981*

[Kok90] : J.J. Kok, "Werktuigkundige regeltechniek II", *lecture notes, Faculty of mechanical engineering, Eindhoven, University of Tecnology, 1990*

[Lee84] : E. Lee, F.R. Grover Jr, "Mathematicl model of the fluidized catalytic cracker plant", *Transactions of the society for computer simulations, Vol. 2, No. 3, pp. 219-236, 1985*

[Lee94] : J.H. Lee, M.Morari, C.E. García, "State space interpretation of Model Predictive Control", *Automatica, Vol.30, No. 4, pp. 707-717, 1994*

[Lju90] : Dag Ljunquist, "Online estimation in non linear state space model with application to catalytic cracking", *Thesis, Norwegian institute of technology, Report no. 90-89-W*

[Mat90] : A. Grace, A.J. Laub, J.N. Little, C. Thompson, "Control system toolbox, for use with Matlab™, user's guide", *The MathWorks Inc., 1990*

[McF90] : R.C. McFarlane, R.C. Reinemann, J.F. Bartee, C. Georgakis, "Dynamic simulator for a model IV Fluid Catalytic Cracking unit", *American institute of chemical engineering 1990, annual meeting, Chicago, Illinois*

[Mor91] : M. Morari, C.E. García, J.H. Lee, D.M. Prett, "Model Predictive Control", *not yet published*

[Pro92] : S.Strand, "Specification of MPC-algorithm PROFIT", *PROFIT, 1992*

[Ric78] : J. Richalet, A. Rault, L. Testud, J. Papon, "Model Predictive Heuristic Control: applications to industrial processes", *Automatica, Vol. 25, No. 3, pp. 335-348, 1978*

[Str91] : S. Strand, "Dynamic optimization in state space predictive control schemes",*Thesis, University of Trondheim, Report 91-11-W, 1991*

Appendix A: Definitions and symbols

Vector:
Denoted as a small letter, like x

$$x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \quad \dim(x) = n$$

Supervector:
Denoted as $x^N(k)$:

$$x^N(k) = \begin{bmatrix} x(k) \\ x(k+1) \\ \vdots \\ x(k+N-1) \end{bmatrix}, \quad \dim(x^N) = Nn,$$

Matrix:
Denoted as capital letters, like A

$$A = \begin{bmatrix} a_{11} & \dots & a_{1m} \\ \dots & \dots & \dots \\ a_{1n} & \dots & a_{nm} \end{bmatrix}, \quad \dim(A) = n \times m$$

or

$$A = \begin{bmatrix} A_{11} & \dots & A_{1m} \\ \dots & \dots & \dots \\ A_{1n} & \dots & A_{nm} \end{bmatrix}, \quad \dim(A_{ij}) = n_1 \times n_2, \ \textit{and} \ \dim(A) = n \times m \times (n_1 \times n_2)$$

$A^i = A^{i-1} \cdot A$ and $A^0 = I$

Vectors and dimensions

Vector	*Dimension*	*Explanation*
x	n_x	state variabeles
u	n_u	manipulated variabeles
d	n_d	measured (or otherwise known) disturbance variabeles
z	n_z	unmeasured (unknown) disturbances (input, parameters)
v	n_v	"model noise" variables
y	n_y	controlled output variables
y_m	n_{ym}	measured output variables
w	n_w	measurement noise variables
r	n_y	references, desired setpoint values
$\overline{d}$	n_d	disturbances when predicted

x_a	$n_{xa} = n_x + n_z$	$x_a = \begin{bmatrix} x \\ z \end{bmatrix}$ *(augmented state vector)*
v_z	n_z	noise variables in the modeling of unknown disturbances
v_a	$n_{va} = n_v + n_z$	$v_a = \begin{bmatrix} v \\ v_z \end{bmatrix}$ *(augmented model noise vector)*
$\bar{x}_a$	n_{xa}	a priori augmented state vector
$\hat{x}_a$	n_{xa}	a posteriori augmented state vector
$\bar{y}$	n_y	a priori model measurement
$\hat{y}$	n_y	a posteriori model measurement
$\bar{\varepsilon}$	n_y	a priori innovation
$\hat{\varepsilon}$	n_y	a posteriori innovation

Matrices and dimensions

Matrix	*Dimension*	*Explanation*
Φ	$n_x \times n_x$	state matrix
Γ_u	$n_x \times n_u$	manipulated variable matrix
Γ_d	$n_x \times n_d$	measured disturbance matrix
Γ_z	$n_x \times n_z$	unmeasured dist. matrix
Ω	$n_x \times n_v$	model noise matrix
C	$n_y \times n_x$	output vector
D_u	$n_y \times n_u$	manipulated variable matrix
D_d	$n_y \times n_d$	disturbance variable matrix
Q_k	$n_v \times n_v$	model noise covariance matrix
R_k	$n_w \times n_w$	measurement noise covariance matrix

Matrix	*Dimension*	*Explanation*
Φ_a	$n_{xa} \times n_{xa}$	$\Phi_a = \begin{bmatrix} \Phi & \Gamma_z \\ 0 & I_{n_z} \end{bmatrix}$ *(augmented state transition matrix)*
Γ_{ua}	$n_{xa} \times n_u$	$\Gamma_{ua} = \begin{bmatrix} \Gamma_u \\ 0 \end{bmatrix}$
Γ_{da}	$n_{xa} \times n_d$	$\Gamma_{da} = \begin{bmatrix} \Gamma_d \\ 0 \end{bmatrix}$
Ω_a	$n_{xa} \times n_{va}$	$\Omega_a = \begin{bmatrix} \Omega & 0 \\ 0 & I_{n_z} \end{bmatrix}$
C_{ma}	$n_{ym} \times n_{xa}$	$C_{ma} = [C \quad 0]$
D_{mu}	$n_{ym} \times n_u$	
D_{md}	$n_{ym} \times n_d$	
K	$n_{xa} \times n_y$	Kalman filter gain
T	$n_{yN} \times n_y$	noise filter used in eq (2.22)

S	$p\times p\times(n_x\times n_x)$	$S=\begin{bmatrix} I_{n_x} & 0 & \cdots & \cdots & 0 \\ \Phi & I_{n_x} & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \Phi^{(p-1)} & \Phi^{(n-2)} & \cdots & \cdots & I_{n_x} \end{bmatrix}$
S_u	$p\times(p+1)\times(n_x\times n_u)$	$S_u=\begin{bmatrix} \Gamma_u & 0 & \cdots & \cdots & 0 & 0 & \cdots \\ \Phi\Gamma_u & \Gamma_u & 0 & \cdots & 0 & 0 & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & 0 & \cdots \\ \Phi^{(m-1)}\Gamma_u & \Phi^{(m-2)}\Gamma_u & \cdots & \Phi\Gamma_u & \Gamma_u & 0 & \cdots \\ \Phi^{(m)}\Gamma_u & \Phi^{(m-1)}\Gamma_u & \cdots & \Phi^{(2)}\Gamma_u & \Phi\Gamma_u & 0 & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & 0 & \cdots \\ \Phi^{(p-1)}\Gamma_u & \Phi^{(p-2)}\Gamma_u & \cdots & \cdots & \Phi^{(p-m)}\Gamma_u & 0 & \cdots \end{bmatrix}$
S_d	$p\times(p+1)\times(n_x\times n_d)$	$S_d=\begin{bmatrix} \Gamma_d & 0 & \cdots & \cdots & 0 \\ \Phi\Gamma_d & \Gamma_d & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ \Phi^{(p-1)}\Gamma_d & \Phi^{(n-2)}\Gamma_d & \cdots & \Gamma_d & 0 \end{bmatrix}$
A	$p\times m\times(n_y\times n_u)$	$A=(C^pS_uL_u+D_u^{p+1}L_u)$
L_u	$(p+1)\times m\times(n_u\times n_u)$	$L_u=\begin{bmatrix} I_u & 0 & \cdots & \cdots & 0 \\ I_u & I_u & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ I_u & \cdots & \cdots & I_u & 0 \\ I_u & \cdots & \cdots & I_u & I_u \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ I_u & \cdots & \cdots & \cdots & I_u \end{bmatrix}$
L_d	$(p+1)\times m\times(n_d\times n_d)$	$L_d=\begin{bmatrix} 0 & \cdots & \cdots & \cdots & 0 \\ I_d & 0 & \cdots & \cdots & 0 \\ I_d & I_d & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ I_d & \cdots & \cdots & \cdots & I_d \end{bmatrix}$
C^p	$p\times p\times(n_y\times n_x)$	$C^p=\begin{bmatrix} C & 0 & \cdots & \cdots & 0 \\ 0 & C & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & \cdots & 0 & C \end{bmatrix}$
D_u^{p+1}	$\{p+1\}\times\{p+1\}\times(n_y\times n_u)$	$D_u^{p+1}=\begin{bmatrix} D_u & 0 & \cdots & \cdots & 0 \\ 0 & D_u & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & \cdots & 0 & D_u \end{bmatrix}$

D_d^p	$p \times p \times (n_y \times n_d)$	$D_d^p = \begin{bmatrix} D_d & 0 & \dots & \dots & 0 \\ 0 & D_d & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & 0 & D_d \end{bmatrix}$

FCC plant variables

Variable	*Definition*	*Workingpoint value*
T_{ril}	Riser outlet temperature	776.2 K
O_d	Regenerator dense bed oxygen fraction	4.2e-03 mole frac.
T_{rg}	Regenerator dense bed temperature	964.5 K
F_{oil}	Mass flow rate of feed to riser	2438 kg/min
T_{oil}	Temperature of feed to riser	420.0 K
F_a	Mass flow rate air to regenerator	1521.1 kg/min
T_a	Temperature of air to regenerator	320.0 K
F_s	Circulation rate of catalyst	17640 kg/min
C_{rc}	Weight fraction coke on regenerated catalyst	4.96e-03 kg/kg

Appendix B: FCC plant

The simulation model of the FCC plant was developed by Lee e.a. [Lee85]. Performing a linearization at a steady state operation point results in a linear model of the form of the equation below. This continuos state space model was discretizised using a Matlab™ discretization routine (see below). The continuos FCC model is of the form:

$$\frac{dx}{dt} = \Phi x + \Gamma_u u + \Gamma_z z$$
$$y = Cx + D_u u + D_z z$$

where the state variable vector is defined as:

$$x = \begin{bmatrix} C_{rc} \\ O_d \\ T_{rg} \end{bmatrix} = \begin{bmatrix} \text{Coke on regenerated catalyst} \\ \text{Regenerator dense bed oxygen fraction} \\ \text{Regenerator dense bed temperature} \end{bmatrix}$$

and the control variable vector:

$$u = \begin{bmatrix} F_{oil} \\ T_{oil} \\ F_a \\ T_a \\ F_s \end{bmatrix} = \begin{bmatrix} \text{Mass flow rate of feed to riser} \\ \text{Temperature of feed to riser} \\ \text{Mass flow rate of air to regenerator} \\ \text{Temperature of air to regenerator} \\ \text{Circulation rate of catalyst} \end{bmatrix}$$

the measurement vector:

$$y = \begin{bmatrix} T_{ri1} \\ O_d \\ T_{rg} \end{bmatrix} = \begin{bmatrix} \text{Riser outlet temperature} \\ \text{Regenerator dense bed oxygen fraction} \\ \text{Regenerator dense bed temperature} \end{bmatrix}$$

the disturbance vector:
$z = k_c^1 = \text{Coke formation rate constant}$

Note that F_{oil}, T_{oil} and T_a are usually not used for the active control of the FCC unit. Consequently, changes in these variables can be considered as disturbances and the matrices Γ_u and D_u should be rearranged accordingly.

The following working point values result from the linearization of the non linear FCC plant, given the values of the model parameters from Lee e.a. [Lee85]

$$x^0 = \begin{bmatrix} 0.4966 & [wt.\%] \\ 0.4200 & [wt.\%] \\ 9.6459 & [\times 100^\circ K] \end{bmatrix}, \quad u^0 = \begin{bmatrix} 2438.0 & [kg/min] \\ 4.2000 & [\times 100^\circ K] \\ 1521.1 & [kg/min] \\ 3.200 & [\times 100^\circ K] \\ 17640 & [kg/min] \end{bmatrix}, \quad y^0 = \begin{bmatrix} 776.2 & [^\circ K] \\ 0.00420 & [wt.] \\ 964.6 & [^\circ K] \end{bmatrix}$$

$$z^0 = 0.0190\ [s^{-1/2}]$$

and from the expressions above the model matrices become:

$$\Phi = \begin{bmatrix} -0.1584 & -0.1594 & -0.1214 \\ -4.4473 & -5.3575 & -4.5891 \\ 0.5139 & 0.5917 & 0.4666 \end{bmatrix},\quad C = \begin{bmatrix} 1.2760 \cdot 10^{+01} & 0 & 5.6070 \cdot 10^{+01} \\ 0 & 1.0 \cdot 10^{-02} & 0 \\ 0 & 0 & 1.0 \cdot 10^{+02} \end{bmatrix}$$

$$\Gamma_u = \begin{bmatrix} -1.2397 \cdot 10^{-05} & 6.8985 \cdot 10^{-05} & 0 & 0 & 5.5132 \cdot 10^{-05} \\ 0 & 0 & 1.4514 \cdot 10^{-03} & 0 & 0 \\ -4.4507 \cdot 10^{-05} & 2.4766 \cdot 10^{-04} & -3.9205 \cdot 10^{-05} & 9.2518 \cdot 10^{-05} & -4.5694 \cdot 10^{-06} \end{bmatrix}$$

$$D_u = \begin{bmatrix} -4.4358 \cdot 10^{-02} & 2.4772 \cdot 10^{-01} & 0 & 0 & 6.1277 \cdot 10^{-03} \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix},\quad \Gamma_z = \begin{bmatrix} 3.5274 \\ 0 \\ 0 \end{bmatrix},\quad D_z = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

This continuos state space model is discretisized with the help of the software program Matlab™. The system control toolbox contains a function "c2dm" which converts the state space model. A sample time of 1 minute was selected. For the algorithm was chosen the tustin algorithm because this algorithm found the discrete state space model most resembling to the continuos state space model.

Appendix C : Quadratic Programming with linear constraints.

To solve the QP problem, first the problem is solved of finding a minimum when no constraints are involved.

$$\underset{x \in R^n}{minimize} \quad F(x) \quad \wedge \quad F''(x) \ exist \tag{C.1}$$

When x^* is the x for which $F(x)$ is a local minimum 2 conditions are fulfilled.

- $\|g(x^*)\| = \|F'(x^*)\| = 0 \quad \Leftrightarrow \quad x^*$ is a stationary point
- $G(x^*) = F''(x^*)$ is semi-positive definite

And looking at the Taylor serie:

$$F(x^* + \varepsilon\, p) = F(x^*) + \varepsilon\, p^T g(x^*) + \frac{1}{2}\varepsilon^{\,2} p^T G(x^* + \varepsilon\theta\, p) p$$

$$\begin{aligned} &where \\ &0 \le \theta \le 1 \\ &\varepsilon : scalar \\ &p : n - vector \end{aligned} \tag{C.2}$$

The problem can be made more complex by introducing linear equality constraints, and is referred to as Linear Equality Problem (LEP):

$$\begin{aligned} &\underset{x \in R^n}{minimize}\ F(x) \\ &subject\ to\ Ax = b \ \ and \ \ dim(A) = t \times n \end{aligned} \tag{C.3}$$

Suppose F has a bounded solution. The imposition of the independent linear constraints reduces the optimization space from n to t-n. The total space can now be divided into 2 subspaces:

- Y : the t independent rows of A.
- Z : the $(n\text{-}t)$ independent rows, the complementary subspace of A.
-

So every n-vector x can be written as the linear combinations of the two subspaces.

$$x = Yx_Y + Zx_Z \tag{C.4}$$

Suppose we have the following solution x^* of the LEP given by:

$$x^* = Yx_Y^* + Zx_Z^* \tag{C.5}$$

Then

$$Ax^* = AYx_Y^* + AZx_Z^* = b \quad \Rightarrow \quad AYx_Y^* = b$$

AY has an inverse, so x_Y^* is uniquely determined.

Until now the nullspace of A remains until now unknown, but has exactly the expected reduction in dimensionality to *n-t.* A basis for x_z would be a subspace of vectors that satisfies $Ap = 0$ (see equation (C.5)). Let that be Z. So AZ=0. Define: $p \equiv Zp_z$.
Optimality of a given feasible point x^*:

$$F(x^* + \varepsilon Zp_z) = F(x^*) + \varepsilon\, p_z^T Z^T g(x^*) + \frac{1}{2}\varepsilon^2 p_z^T Z^T G(x^* + \varepsilon\theta p) Zp_z$$

where (C.6)

$0 \le \theta \le 1$

ε: *positive scalar*

If the term $p_z^T Z^T g(x^*)$ vanishes for every $p_z \Leftrightarrow Z^T g(x^*) = 0$

This is a necessary condition for $x^*$to be a local minimum. $Z^T g(x^*)$ is called the projected gradient of F at x^*. The above result implies that $g(x^*)$ must be a linear combination of the rows of A:

$$g(x^*) = \sum_{i=1}^{m} a_i \lambda_i^* = A^T \lambda^*$$

(C.7)

λ^*: *vector of Lagrange multipliers*

Now:

$$F(x^* + \varepsilon Zp_z) = F(x^*) + \frac{1}{2}\varepsilon^2 p_z^T Z^T G(x^* + \varepsilon\theta p) Zp_z \qquad \text{(C.8)}$$

If $Z^T G(x^*)Z$ is indefinite, $F(x^*)$ is strictly lower than in an arbitrary space round x*. Thus $Z^T G(x^*)Z$ (projected Hessian matrix) must be semi positive definite.

So the local minimum has 3 conditions to fulfill:

- $Ax=b$
- $Z^T g(x^*) = 0 \Leftrightarrow g(x^*) = A^T \lambda^*$
- $Z^T G(x^*)Z$ is semi positive definite

The complexity of the problem increases again by introducing the Linear Inequality Problem (LIP).

$$\underset{x \in R^n}{\text{minimize}}\ F(x)$$
$$\text{subject to } Ax \ge b \qquad \text{(C.9)}$$

Define: $\hat{A}x = b$ as the equality constraints of the LIP problem.

At a feasible point $\hat{x}$, there are 3 different possibilities for the constraints:

- constraint active: $a_i^T \hat{x} = b_i$
- constraint inactive: $a_i^T \hat{x} > b_i$
- constraint violated: $a_i^T \hat{x} < b_i$

Suppose that the i_{th} constraint is active at $\hat{x}$: $a_i^T \hat{x} = b_i$. There are 2 feasible search directions with respect to an inequality constraint.

- $a_i^T p = 0$: binding perturbation
- $a_i^T p > 0$: non-binding perturbation

The conditions stated in the previous problem are not sufficient to guarantee a local minimum, because there can exist a non-binding perturbation p that is in a descent directory F. Hence, an extra condition is added.

$$forall\ p: Ap \geq 0 \quad \Rightarrow \quad g(x^*)^T p \geq 0 \tag{C.10}$$

Since it is known that: $g(x^*) = \hat{A}^T \lambda^*$

$$\begin{aligned} &g(x^*)^T p = \lambda_1^* a_1^T p + \ldots + \lambda_t^* a_t^T p \geq 0, \\ &where: \\ &a_i^T p \geq 0, i = 1 \ldots t \end{aligned} \tag{C.11}$$

holds only if $\lambda_i^* \geq 0, i = 1, \ldots, t$

This results in the necessary conditions:

- $Ax^* \geq b$ with $Ax = b$
- $Z^T g(x^*) = 0 \quad \Leftrightarrow \quad g(x^*) = A^T \lambda^*$
- $\lambda_i^* \geq 0, i = 1, \ldots, t$
- $Z^T G(x^*) Z$ is semi positive definite

Now a general set of conditions has been found for which a feasible point x must be a local minimum. Active set methods for linear inequality constraints are used (LIP) to find this point.

$$\begin{aligned} &\underset{x \in R^n}{minimize}\ F(x) \\ &subject\ to\ Ax \geq b \quad \dim(A) = m \times n \end{aligned} \tag{C.12}$$

Assume t constraints are active at x^* and define matrix $\hat{A}$ whose i_{th} row contains the coefficients of the i_{th} constraint:

$$g(x^*) = \hat{A}^T \lambda^*, \lambda^* \geq 0 \tag{C.13}$$

If the correct active set was known a priori, the solution of LIP would be a solution of the equality constrained problem (LEP). This problem can easily be solved. A "working set" is selected from the LIP. And the ideal working set is the correct active set. This ideal working set can be found in 2 phases:

1. determination of a feasible point that exactly satisfies a subset of the constraints $Ax \geq b$.(one changes the subset until a feasible point is found and no constraints are violated)
2. generation of an iterative sequence of feasible points that converges to the solution of LIP

The Quadratic Programming problem solved in MPC is a special form of the LIP problem, where the projected Hessian matrix $Z_k^T G Z_k$ is known a priori to be positive-definite at every iteration. This simplifies the methods of finding a minimum.