NTNU
Norwegian university of science and technology

Faculty of natural science and technology
Depertement of chemestry engineering

PROJECT WORK 2007

Comparison of control structures for maximizing throughput

Théogène Uwarwema

Abstract

In this project four different control structures were compared to maximize the throughput for a process model. The control structures considered were single loop control, single loop with feedforward control on level controllers, cascade control and model predictive control.
The process model was built and simulated within Simulink/Matlab software. One simulation scenario was applied to all four control structures, this for a consistency comparison of their responses on eventually disturbances into the process model.
Single loop control is naturally used to stabilize the process in control structure hierarchy. The single loop - PI controller used in this project was tuned using SMIC tuning rules, obtained results were realistic. However, disturbances rejection was very poor using this control structure. The single loop-feedforward control on level controllers improved the regulatory control significantly.
For the cascade control structure, using extra measurements resulted in good rejection of local disturbances on secondary variables. But disturbances downstream those extra measurements were poorly rejected. Model predictive control was built on the top of the single loop structure and implemented using the Simulink/Matlab inbuilt MPC controller. The results obtained using MPC controller were superior to those obtained using the single loop and cascade control structures. Whereas they were nearly the same as those obtained using single loop with feedforward control scheme. Still tuning the MPC controller was not a trivial task as the tuning parameters are mostly a matter of "rules of thumb", based largely on experience gained from simulation of typical problems.

I want to thank my supervisor professor Sigurd Skogestad at the Department of Chemical Engineering and PhD-student Elvira Marie B. Aske for their valuable guidance, support and advice. I would also like to express my gratitude to PhD-student Henrik Manum and post.doc. Eduardo S. Hori for their help with Simulink.

Trondheim, Norway, November 2007

Théogène Uwarwema

Contents

1. Introduction

The plant optimum can in many cases be simplified to maximum throughput. Assuming sufficiently high product prices, low feed and utilities cost; the maximum throughput is realized with maximum flow through the bottleneck (Aske et al., 2007).
The production rate is commonly set at the inlet to the plant, with inventory control in the direction of flow (Price et al., 1994). With this assumption we typically fix the feed rate. However, the feed rate is usually a degree of freedom while operating a plant, and very often the economic conditions impose to maximize the production rate; that imply an increase of the feed rate. Conversely as the feed rate increase one will eventually reach a constraint F_{max} of a flow variable F, which becomes a bottleneck for the further increase in the feed rate. Consequently, maximum flow through the bottleneck can usually not be achieved in practice due to hard constraints, which can not be violated freely.

Then to allow the plant operational feasibility one needs to reduce the feed rate and "back off". On the other hand this option gives an economic loss; therefore the back off needs to be minimized. To achieve minimum back off the throughput manipulator (TPM) should be located so that controllability of the bottleneck unit is good (Skogestad, 2004).
The feedrate (TPM) should be selected as a direct bottleneck manipulator as it avoids back and directly maximizes the flow through the bottleneck (Price et al., 1994).

In this project we compare four different control structures to maximize the plant throughput.

1. *Single loop control*, when the bottleneck does not move one can use a single loop PI-controller on the throughput manipulator (Skogestad, 2004).
2. *Single loop with feedforward control,* for situations where single loop control by itself is not satisfactory, significant improvement can be achieved by adding feedforward control.
3. *Cascade control* is a special case, where we introduce extra measurements to tightly control the secondary outputs, this handles local disturbances and reduces the back off on the primary controlled variables, and thus it maximizes the throughput.
4. *Multivariable control*, multivariable constrained control has the advantages that interactive processes are coordinately controlled and there is no logic needed to handle changing constraints and smooth transition between active constraints. Model predictive control (MPC) is used in this project. To use multivariable control one needs a multivariable dynamic model, here we use volumes as an additional dynamic degree of freedom.

This project is organized as follows. We begin by building the model in Simulink/Matlab, and then implement four different control structures: single loop control, single loop with feedforward control, cascade control, and MPC within the Simulink/Matlab inbuilt MPC controller block. Thereafter the model is tuned and simulated using Skogestad's tuning rules (SIMC) for the single loop and cascade control structures. The MPC controller is tuned based on other tunings parameters than SIMC. Finally, we compare and discuss results obtained from those four control structures.

2. Process model

2.1. Physical realization

From figure 2.1, we consider a process model consisting of two units in series. We assume that the bottleneck is fixed and located on the output flow (F_4) of unit two. The process model has one feed flow F_0 which enters the process through unit one, F_2 is the output flow of unit one and the input (feed) for unit two, F_4 is the process output flow thus the process product. Between the unit one and its buffer tank there is flow F_1, whereas flow F_3 is between unit two and its buffer tank. The process has four disturbances (denoted d), disturbance d_1 enters the process through F_0, d_2 through F_1, d_3 through F_2 and d_4 through F_3 see Figure 2.1.
The main objective here is to maximize the process throughput using four different control structures, single loop control, single loop with feedforward control on level controllers, cascade control and model predictive control. The structure presented in figure 2.1 is a single loop control.

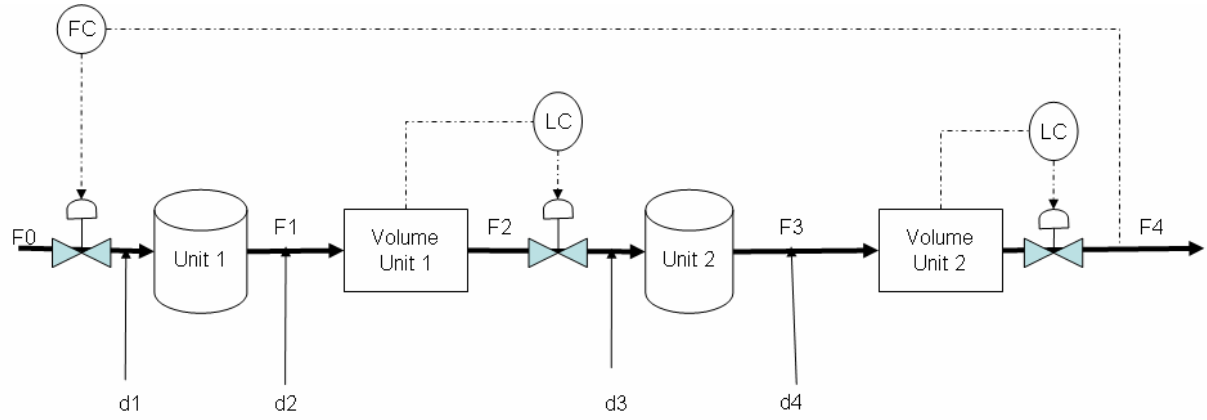

Figure 2.1 The physical process model with two units

The process model above has three valves, one on the feed, the second on the output flow of unit one F_2 and the third on the output flow of unit two F_4. Valves on flows F_2 and F_4 are used to successively control the level in unit one and unit two. The last valve on the feed flow F_0 is used to control the production rate F_4, see figure 2.1.

2.2. Building the model in Simulink

Simulink 6.6 R2006a is used to build the Simulink model of the physical model in figure 2.1. The Simulink model shown in figure 2.2 is made of different blocks taken from the Simulink library by drag and drop. Unit one is represented by the block named transfer fcn1, unit two by transfer fcn2, level in the units are illustrated with integrator1 and integrator2 blocks successively for the level in unit one and unit two. Disturbances are sent into the process as steps and they are represented by step blocks (red color). The floating scope (green) allows us to visualize the controlled variables response, before we eventually can save the results for later use if satisfy. The “to workspace” block gives us the possibility of plotting our results and saving those as a Matlab file.

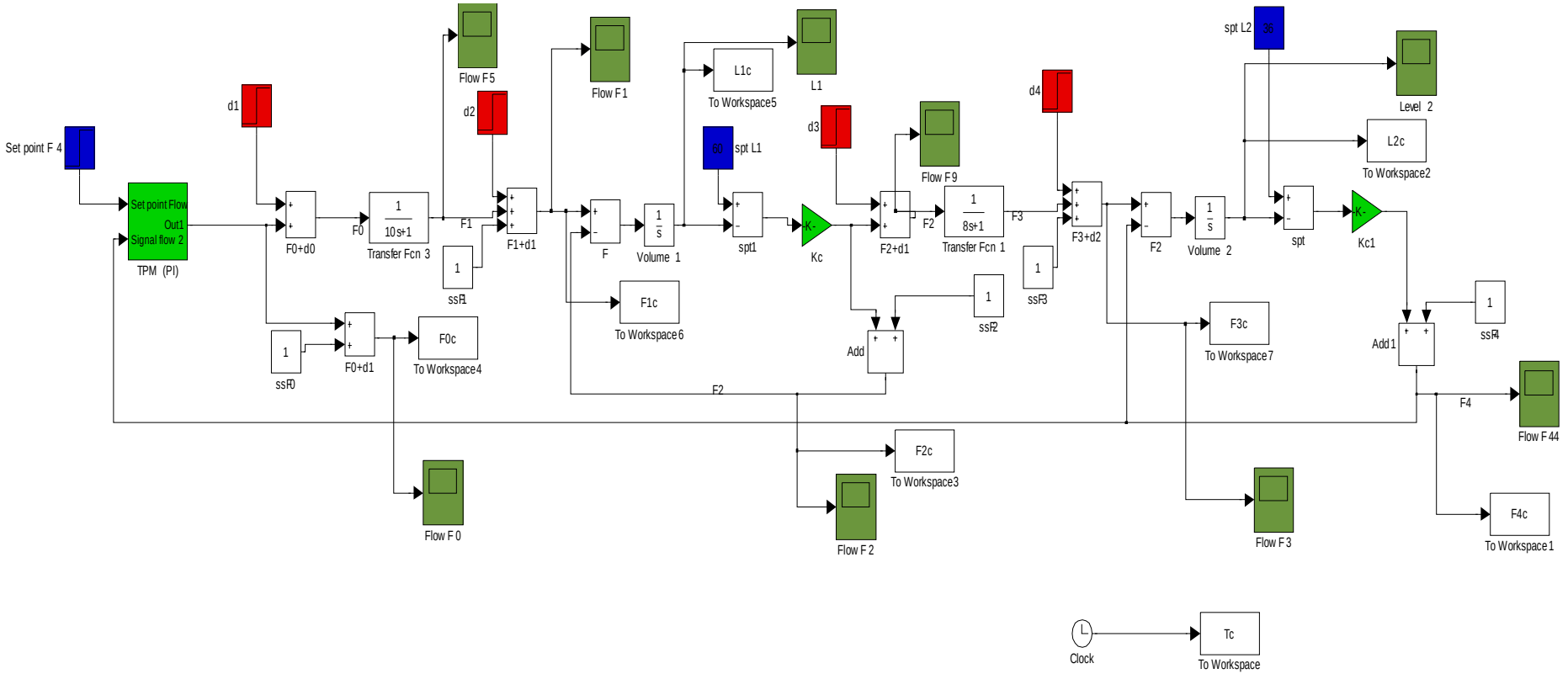

Figure 2.2 Simulink model with two units including single loop control (PI, P)

The later option allows us to manipulate the simulation results without making any further simulation run. Blocks named Kc, Kc1 and TPM are controllers blocks (green). Set-points for the controlled variables are illustrated by the blue blocks.

Both unit one and unit two are of first order model as well as their buffer tanks; disturbances are assumed to be constant. The feed flow rate, disturbances, time constants for unit one, unit two, buffer tank one and two are stated in table 2.1.

Table 2.1 Time constants, feed rate and disturbances used in the model presented in figure 2.2

Variables	**Values**	**units**
τ (unit 1)	10	min
τ (volume 1)	20	min
Set-point volume 1	60	m^3
τ (unit 2)	8	min
τ (volume 2)	12	min
Set-point volume 2	36	m^3
Feed flow rate F_0	1	m^3/min
d_1	0.5	m^3/min
d_2	0.5	m^3/min
d_3	0.5	m^3/min
d_4	0.5	m^3/min

The overall transfer function from F_0 to F_4 is given by:

$$F_4 = \left(\frac{1}{10s+1}\right)\left(\frac{1}{20s+1}\right)\left(\frac{1}{8s+1}\right)\left(\frac{1}{12s+1}\right)F_0 \qquad (1)$$

2.3. Control targets

The control targets in the process model presented in figure 2.2 are levels in unit one and two (L_1 and L_2) and the output flow F_4. The production rate is set at the inlet to the process model and thereby adjusted with the feed flow. While outflows from unit one and unit two are used for level control within these units. We want to maximize the plant throughput; a natural way to achieve this is to increase the feed flow rate into the process model. Due to the plant operational constraints and disturbances, we have to back off from the maximum production rate F_{4max}, this to avoid the dynamic infeasibility. The back off is given by

$$b = F_{4\max} - F_{4s} \qquad (2)$$

The control objective here is to minimize the back off; allowing the plant operational feasibility and ensuring the optimal economic conditions. In this case the back off is determined by the dynamic variation in the flow F_4. An improved bottleneck control will hold the back off constant. In this project the disturbances are assumed to be known, the back off b is adjusted according to the expected disturbances and the goal is to get the production set-point F_{4s} closer to F_{4max}.

Furthermore volumes are used as buffer tanks in expectation to damp disturbances, smooth the dynamic variation and with that reduce the back off in the flow F_4. They are used as range control in single loop and cascade control structure and as dynamic degree of freedom in model predictive control structure.

3. Control structure for maximizing throughput

We consider the process model given in figure 2.1. In all cases we assume that the bottleneck is located in the flow F_4.

3.1. Single loop-PI controller

When a simple control structure is desired, it is wise to pair variables that are close to each other physically. This means that when one wants to control some variable in the outlet stream of a distillation column, the manipulated variable should probably be one directly related to the distillation column (for example feed flow rate or feed temperature).
The presented model has a fixed bottleneck with feed rate (F_0) as the manipulated variable (u) and the bottleneck flow F_4 as the controlled variable (y). In this control structure we use single loop controllers for the overall feed flow and the levels in the two units. For the TPM we use PI-controller to control the output flow F_4, while levels in unit one and two are controlled by a single proportional controllers.
Assuming nominal volume $\hat{v}$ such that holdup time is $3\tau_{eff}$ where τ_{eff} is the effective time constant; the required volume is given by $\hat{v} = \frac{3}{K_c}$, where K_c is $K_c = \frac{1}{\tau_{eff}}$ and is the proportional controller tuning parameter. A proportional controller can not hold the level at its set point that means an increase in the inflow, will result to an increase in the buffer volume and the opposite for a decrease in the inflow. To hold the level at their nominal values one needs an integral action. The last option is not so necessary (desired) in this project, because we want buffer tanks to vary and damp eventual disturbances.

Both P and PI are tuned by using SIMC tuning rules (Skogestad, 2003). A first order plus time delay transfer function G_s (3) is obtained from the overall transfer function (1) using half rule.

$$G_s = \frac{1}{26s+1} e^{-24s} \tag{3}$$

The closed loop time constant τ_c is chosen to be $\tau_c = \theta$ for tight bottleneck control and $\tau_c = 3\theta$ for a smooth control.

Using $k_c = \frac{1}{k} \frac{\tau_1}{\tau_c + \theta}, \tau_I = \tau_1$ and the first order plus time delay transfer function given in (3) the tuning parameters for the PI controller was calculated and are stated in table 3.1

Table 3.1 tuning parameters for the TPM in figure 2.2

Tuning parameters	$\tau_c = \theta$	$\tau_c = 3\theta$
k_c	0.54	0.27
τ_I	26	26

Tuning parameters for levels in our system are found using $k_c = \frac{1}{\tau_{eff}}$ see table 3.2

Table 3.2 tunings parameters for level controllers for the model in figure 2.2

Parameters	Volume 1	Volume 2
k_c	0.05	0.083

Depending on the time delay and where the disturbances are situated in the process as well as their importance from economical view point. One may evaluate if a single loop is adequate or if one can introduce extra measurements for an improved tight bottleneck control.

3.2. Single loop-feedforward control

In section 3.1 it was emphasized that single loop control is an important control structure that is widely used in the process industries. However single loop control has certain inherent disadvantages as no corrective action is taken until after the deviation in the controlled variable occurs, it may not be satisfactory for processes with large time constant and/or long time delay.
Significant improvement can be achieved by adding feedforward control for situation in which single loop is not satisfactory. But feedforward requires that the disturbances be measured or estimated on line. In this project additional to the single loop control (figure 2.1) we implement the feedforward in the control structure. The configuration is shown in figure 3.1, where the output of the feedforward control and the single loop controllers are added together and combined signal is sent to the control valve.

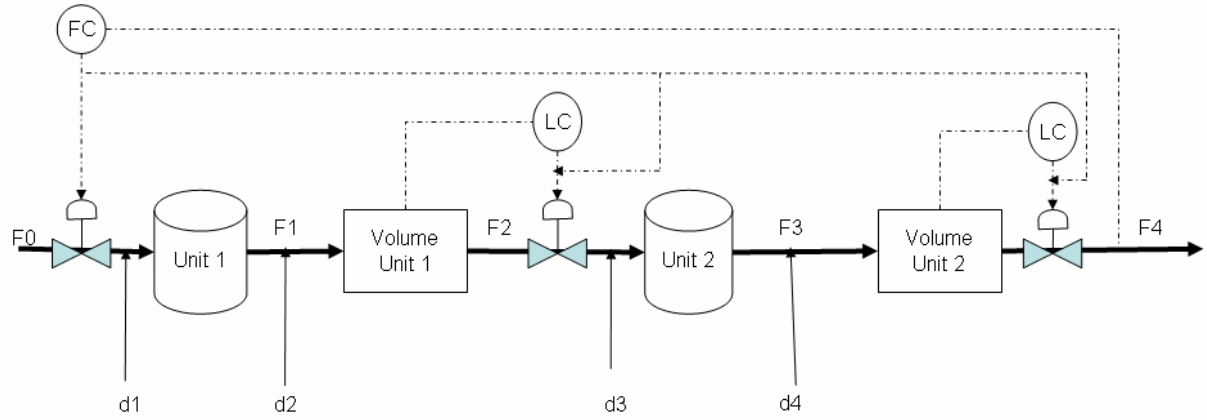

Figure 3.1 Feedforward-single loop control, the signal from the TPM (F0) is added to the level controllers

The control scheme in figure 3.2 can provide better control of the outputs flow F_4. The main idea here is to measure important disturbance variables and take corrective action before they upset the process. The signal from the TPM (F0) is directly added to the level controllers upstream the bottleneck. This reduces the effective time delay and can give a quite good

exploitation of buffer volumes and thus damp disturbances. Furthermore, good disturbance rejection results in minimum back off, and thereby maximum production.

The Simulink model for this control scheme is shown figure 3.3. Here we use a special type of feedforward control called *ration control*. Its option is to maintain the ratio of two process variables at a specified value (Seborg et al., 2004). In figure 3.3 variable F2 and F4 are compared to the feed flow rate out of the TPM (F0). Here the ratio is one in both cases, as F_0, F_1, F_2, F_3, and F_4 are assumed to equal at steady state.
Level controllers are tuned using the SIMC tuning rules as it's done in single loop control. The TPM is tuned using tuning constants for a typical flow PI controller with $\tau_I = 0.5$ min. and $K_c = 0.2$ as tuning parameters. For further reading on feedforward tuning see (Seborg et al., 2004).

3.3. Cascade control

For cascade control the output from one controller is the input to another. Here we use extra measurements to reduce the back off in F_4. From figure 2.2 in addition to F_4 control, we introduce a secondary loop on F_2 (For other extra measurements see appendix B). The reason of controlling F_2 is to handle disturbances d_0 and d_1 before they attend F_4.

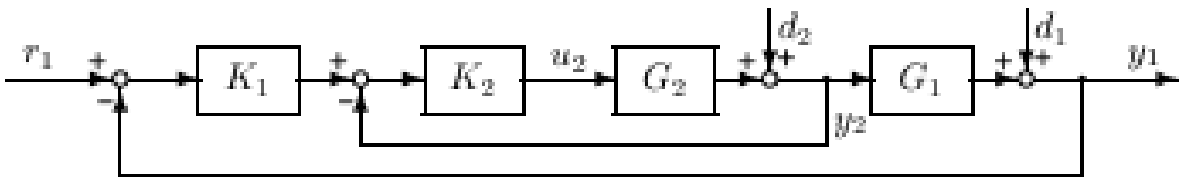

Figure 3.2 common case of cascade control where the primary output $y1$ depends directly on the extra measurement $y2$ (Skogestad and Postlethwaite, 2005)

Again using The SIMC tuning rules, where the idea is to tune the controllers such that the resulting transfer function from r to y is $T = \frac{e^{-\theta s}}{\tau_c s + 1}$ where θ is the effective delay in G (from u to y) and τ_c being the tuning parameter selected for fast control. This approach is applied to the cascaded model in figure 3.2. The inner loop K_2 is tuned based on G_2, one then get $y_2 = T_2 r_2$ where $T_2 \approx \frac{e^{-\theta_2 s}}{\tau_{c2} s + 1}$ and θ_2 is the effective delay in G_2. The inner loop is fast; its response may be approximated time delay for the tuning of the slower outer loop K_1

$$T_2 \approx 1.e^{-(\theta_1 + \tau_{c2})s} \qquad (4)$$

The model for tuning of outer loop (K_1) is

$$\tilde{G} = G_1 T_2 \approx 1.e^{-(\theta_1 + \tau_{c2})s} \qquad (5)$$

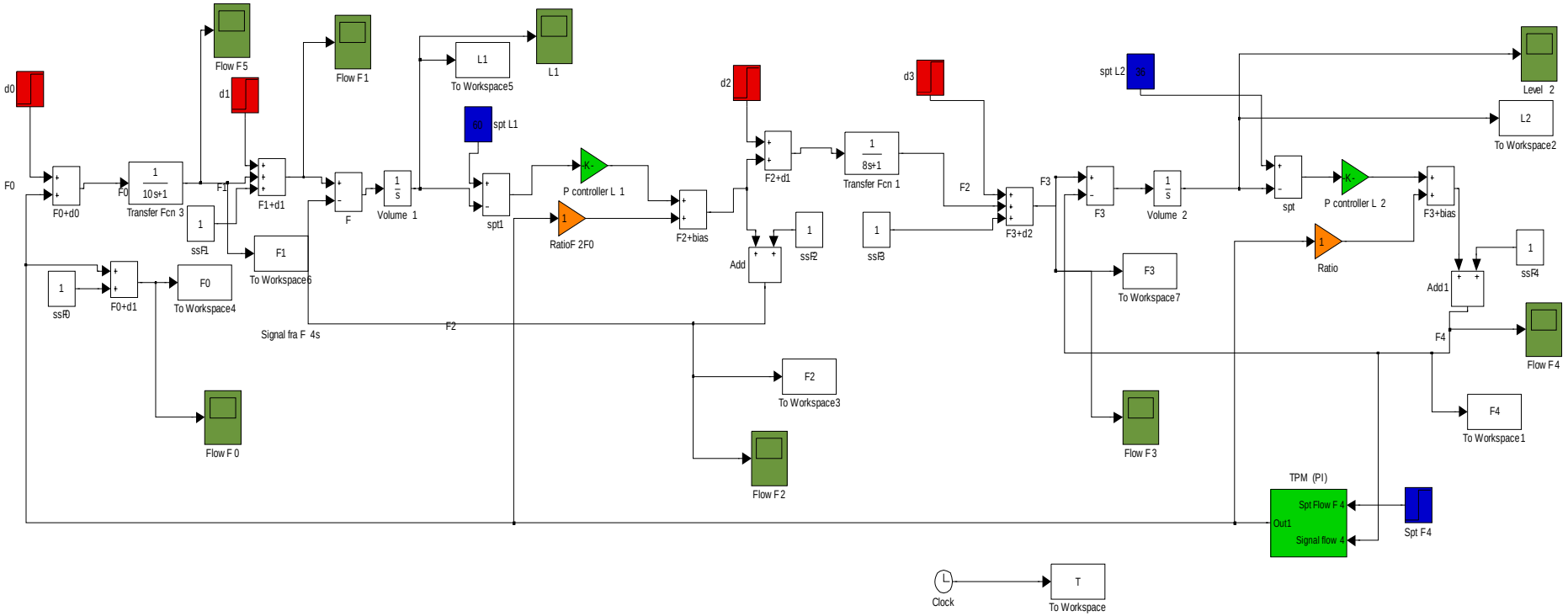

Figure 3.3 The process model with single loop-feedforward (ratio) control. Orange blocks illustrate the ratio control

Applying equation (4) and (6) on the model in figure 3.4, the models used to tune the inner loop (TPM2) and the outer loop (TPM1) are given by:

$$G_I = \frac{1}{25s+1}e^{-5s} \qquad (6)$$

and

$$G_O = \frac{1}{12s+1}e^{-16s} \qquad (7)$$

The tuning parameters for the models in (6) and (7) are stated in table 3.3

Table 3.3 tuning parameters for cascade control in figure 3.4

Parameters	$\tau_c = \theta$	$\tau_c = 3\theta$
G_2 (The inner loop)		
k_c	2.5	1.25
τ_I	25	25
G (The outer loop)		
k_c	0.38	0.19
τ_I	12	12

Generally the objective of the regulatory layer is to locally control secondary measurements so that the effect of the disturbances on the primary measurements can be handled by the above layer (Skogestad, 2004). Multivariable control (e.g. MPC) is suited for this and in the hierarchy control it follows the regulatory layer.

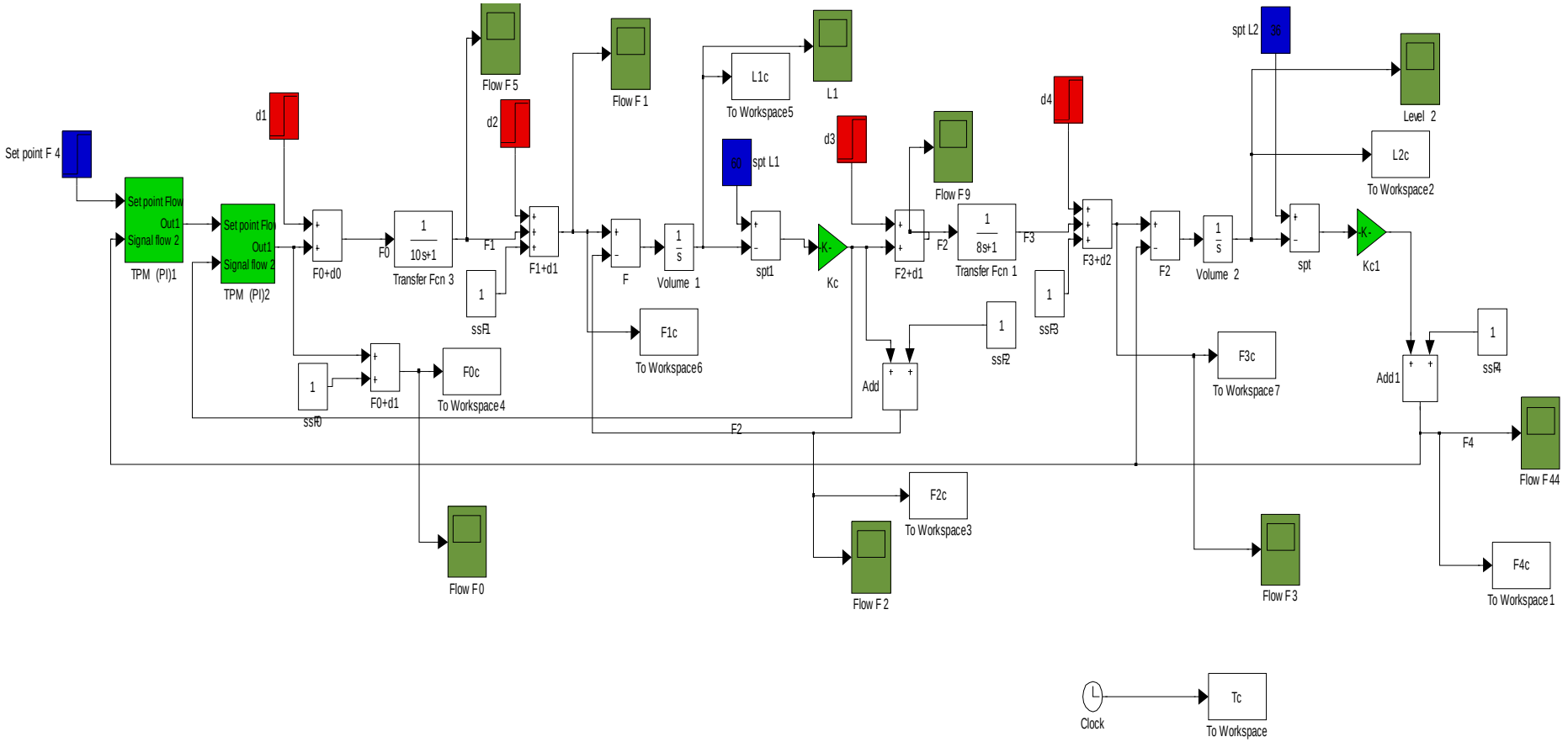

Figure 3.4 Cascade control with measurement on F_2 and F_4

3.4. MPC controller

The MPC controller used in this project was implemented in the inbuilt MPC Simulink library within Matlab. The MPC toolbox is described in appendix A.

3.4.1. Building the MPC controller

The MPC controller is built on the top of the Simulink process model in figure 2.2 using model predictive controller toolbox. Generally the MPC block controller lets us design, simulate and tune model predictive controllers. The MPC design tool is used to create a new controller or modify an existing one. Measured disturbance and reference signals are external inputs to the MPC block by default. We can choose whether to load these from the workspace or from the block port input. In this project we use the latter option see figure 3.2.

The MPC Controller Block in figure 3.5 receives the current measured output (MO) $[F_4\ L_1\ L_2]^T$, reference (set-point) signal $[F_{4s}\ L_{1s}\ L_{2s}]^T$, and measured disturbance signal (no MD here), and outputs the optimal manipulated variables (MVs) $[F_0\ sptL_1\ sptL_2]^T$ by solving a quadratic program. The block is based on an MPC object, which provides performance and constraint specifications, prediction model, weights, as well as the sampling time of the block. Before we begin to design the MPC controller, we have to assign the linearization points I/O on the MO and MV. Thereafter a double click on the MPC controller block in the model presented in figure 3.2 gives the MPC controller mask.

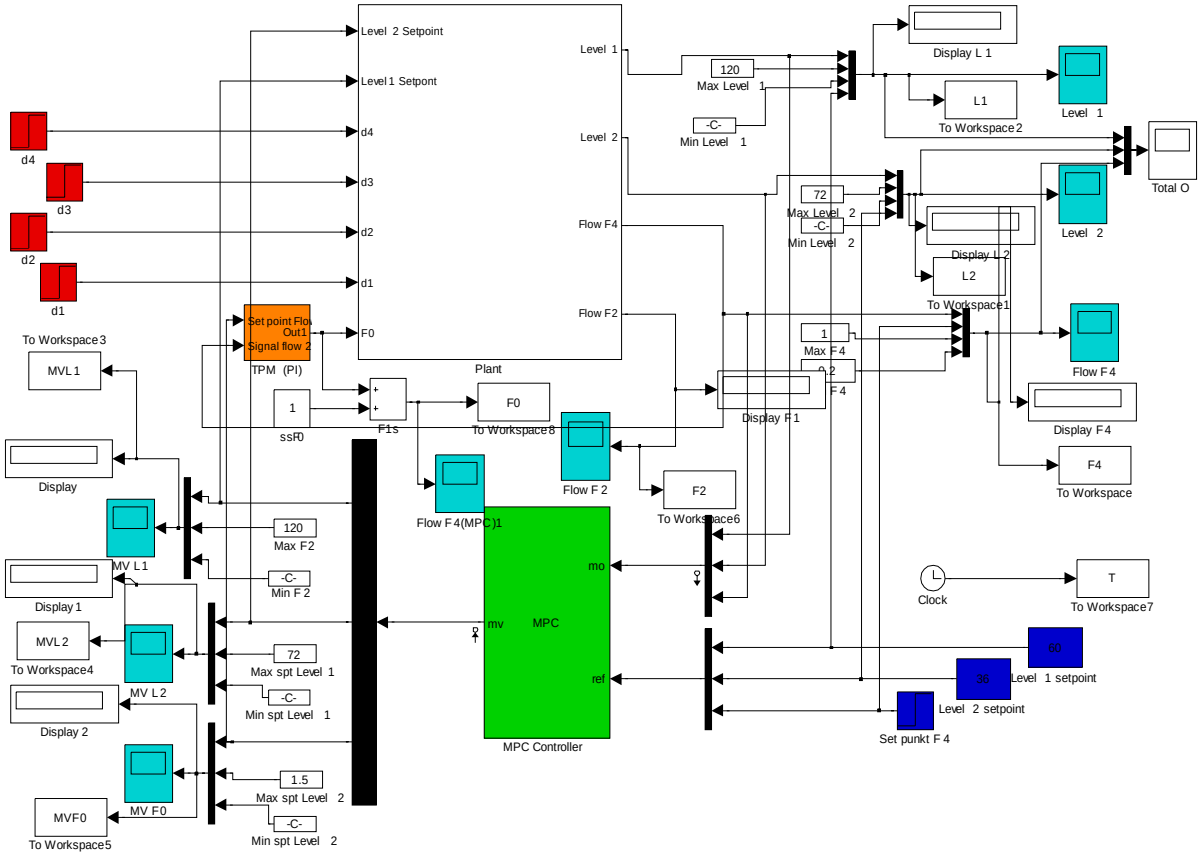

Figure 3.5 The Simulink process model with MPC controller (green) on the top of single loop PI- control structure (orange)

The mask requires that we specify a valid MPC controller object. There are two ways of providing an MPC controller object:
The first is to load an existing MPC object from the workspace. This means that the MPC object is implemented in Matlab software and then exported to the workspace.
The second is that one has to click the design button from the controller mask to open the MPC design Tool and design the MPC controller object there.

The latter options is used in this project, by clicking the Design button in the mask one has to specify how many manipulated variable are in the system, in this project we have three MVs, which are the feed flow F_0 and the set-points of the two levels in our process model; $[F_0\ sptL_1\ sptL_2]^T$. After specifying the MVs, the MPC tool design begins the operations to build the controller. When these operations are finished, we have a linearized discrete-time; state model of the form presented in equation (8) – (9) and a default MPC controller has been constructed. The MPC controller is ready to be adjusted.

$$x_{k+1} = Ax_k + Bu_k \tag{8}$$

$$y_k = Cx_k \tag{9}$$

Where x is the state vector, u the input vector; y is the controlled outputs vector.

We can run closed-loop simulations while we are editing the MPC controller in the MPC design Tool. In this case, the controller parameters used for simulating the Simulink diagram are those specified in the MPC design Tool, so that we can easily tune the parameters of the controller. One has to remember to export the MPC controller to the workspace, before closing the MPC design toolbox. Or eventually save it as a Matlab file for later use. Then it can be loaded as an MPC object to the workspace.

When one needs to switch between MPC control and another type of control (e.g., manual control, PI-controller in this case) during a simulation, the enabling of input port for externally manipulated variables to the plant is useful. When this input port is enabled, the block is resized and a new input signal to the MPC controller appears which represents the actual manipulated variables in the process. The MPC algorithm to update the internal state estimate then uses those new inputs, rather than the manipulated variables generated by the MPC controller (see appendix A).

Using the same approach one can enable or disable measured disturbances. If they are disabled the block has two input signals, namely measured outputs and references. When they are enabled, the block has measured disturbances as the third input signal (see appendix A).

3.4.2. MPC controller tuning

Basic formulation of MPC

For the basic formulation of MPC we suppose that the cost function (10) is quadratic, that the model is linear, and that constraints are on the form of linear inequalities; we also assume that everything is time-invariant. Additionally, we assume that the cost function does not penalize particular values of the input vector $u(k)$, but only changes of the input vector $\Delta u(k)$.

Furthermore, we do not assume that the state variables can be measured, but that their estimate $\hat{x}(k/k)$ of the state $x(k)$ can be obtained, the notation indicates that this estimation is based on measurements up to time k that is on measurement of the outputs up to $y(k)$, and on knowledge of $u(k-1)$, since the next input $u(k)$ has not yet been determined.
$\hat{x}(k+i/k)$ and $\hat{y}(k+i/k)$ $(i=0,1,...H_p-1)$ denote the prediction made at time k, of the variables x and y at time $k+i$, assuming that some sequence of inputs $\hat{u}(k+j/k)$ $(j=0,1,...i-1)$ will have occurred. These predictions are made consistently with the assumed linearized model (8) – (9).

$$V(k)=\sum_{i=H_W}^{H_P}\left\|\hat{y}(k+1/k)-r(k+i)\right\|^2_{Q(i)}+\sum_{i=0}^{H_u-1}\left\|\Delta\hat{u}(k+i/k)\right\|^2_{R(i)} \qquad (10)$$

From MPC controller building we saw that the MPC block outputs optimal manipulated variables by solving quadratic program. The cost function (10) penalizes the deviations of the predicted controlled outputs $\hat{y}(k+i/k)$ from a reference trajectory $r(k+i/k)$. The notation indicates that this reference trajectory may depend on measurements made up to time k.
The prediction horizon in (10) has length H_p, but it does not mean that we start to penalize deviations of y from r immediately. The reason is that there may be some delay between applying an input and seeing any effect. H_u is the control horizon. We will always assume that $H_u \le H_p$, and that $\Delta\hat{u}(k+i/k)=0$ for $i \ge H_u$ so that $\Delta\hat{u}(k+i/k)=\Delta\hat{u}(k+H_u-1/k)$ for all i ≥ H_u, see figure 3.6.

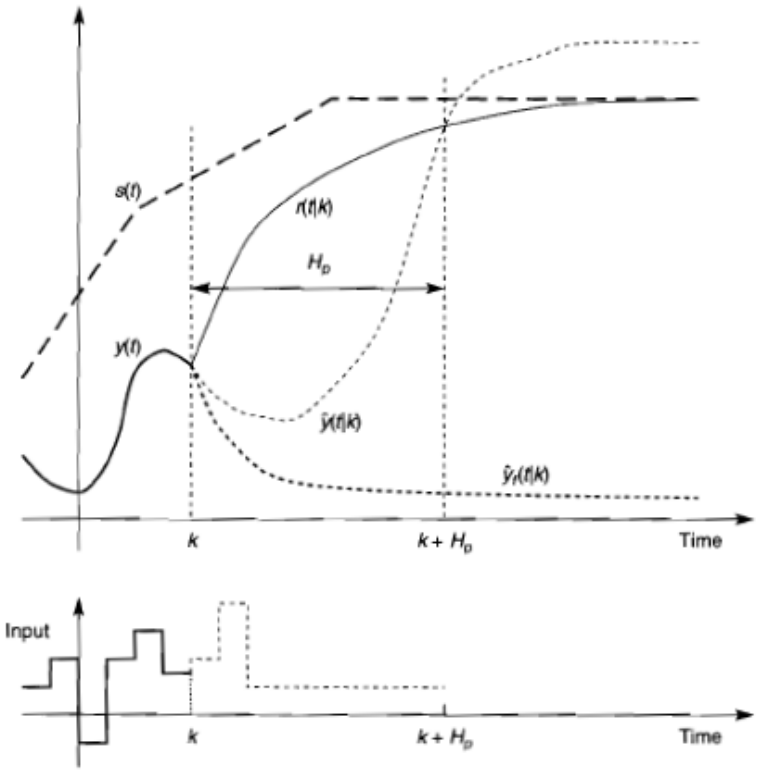

Figure 3.6 Predictive control

$Q(i)$ and $R(i)$ are the weighting matrices, to unsure $V(k) \geq 0$ we need $Q(i)$ and $R(i) \geq 0$ .The cost function (10) penalizes changes in input vector, but not its values. In some cases, one may need to penalize deviations of the input vector from some ideal resting value: then an additional term of the form $\sum \|\Delta U(k+i/k) - u_0\|_S^2$ is added. This is done only when there are more inputs than variables which are to be controlled to set-points (Maciejowski, 2002). In this project we shall not include such a term as we do not have more inputs than controlled variable to set-points.

Weights $Q(i)$ and $R(i)$, the prediction and control horizon H_p and H_u, the window parameter H_W, and the reference trajectory $r(k+i)$, all affect the behavior of the closed-loop combination of plant and predictive controller. Principally the weights may be dictated by the economic objectives of the control system, but usually they are in effect tuning parameters which are adjusted to give satisfactory dynamic performance.
In the following we shall examine the effect of these, try to obtain insight into the effects of the parameters on the process model presented in figure 3.5, and some systematic methods of adjusting them. Two MPC controllers, MPC1 and MPC2 are implemented and investigated in this control scheme.

Prediction and control horizon

In this project we choose $H_u \leq H_p$, we suppose that the plant includes a pure time delay equivalent to D sampling instants. This means that the controllers move, u_k, has no effect until $y_k + D + 1$. In this case it is essential that $H_p >> D$ and $H_u << H_p - D$ as this forces the controller to consider the full effect of each move.

Table 3.4 Sample time control and prediction horizon for MPC controllers

Variables	MPC2	MPC3
Sample time [min]	1	1
Prediction horizon	120	120
Control horizon [blocking]	[3 5 10 15 20 25 30]	[3 5 10 15 20 25 30]

Prediction horizon H_p and control horizon H_u are chosen to be 120 and 80. As the control horizon H_u increase, the MPC controller tends to become more aggressive and the required computational effort increases. However we can reduce this computational effort by input blocking. Clicking on the model and horizon option and then select blocking check box gives as this possibility. This deactivates the control horizon and computes moves as they are defined in blocking options (see table 3.4).
The prediction horizon H_p is often selected by using a thumb rule to be $H_p = H_u + D$ so that the full effect of the last input move is taken into account. Using a lower value of H_p tends to make the controller more aggressive.

Weight tuning

The weight tuning tab let us tune the output $Q(i)$, the input $R(i)$ weighting matrices and the overall slider control. The last named adjusts the weights on all variables simultaneously, a large value gives fast response whereas a lower value gives a more robust response. The overall slider control can be adjusted between 0 and 1. The overall slider control was adjusted to 0.8 in both controllers studied here.

Inputs weight

We have two options for weighting of manipulated variables:

The weight column penalizes deviations on each manipulated variable from its nominal value. A larger weight value keeps its corresponding manipulated variables closer to its nominal value, but this can result steady state error (offset) in the output variables unless one has extra MVs at his disposal. In this case we would like to hold F_0 near its nominal value and let sptL1 and sptL2 vary freely.

The Rate Weight column penalizes changes on MV. The simultaneous objective is to minimize the weighted sum of controller adjustments.
An increase of penalty on a given MV causes the controller to change it more slowly. From table 3.5 we hardly penalize changes in F_0, and allow sptL1 and sptL2 vary quite freely by setting their rate weight smaller. Contrary to weight, the rate weight values have no effect in steady state.

Table 3.5 Weight on manipulated variables

Variables	Weights MPC2	Weights MPC3	Weights rate MPC2	Weights rate MPC3
F0	1e-10	1e-10	400	1.50
SptL1	1e-20	1e-20	1e-4	1e-4
SptL2	1e-20	1e-20	1e-4	1e-4

Output weight

The output weight tuning parameters let us dictate the accuracy within the desired output. It states the accuracy with which each output must track its set-point (or reference). This means that the controller predicts deviation over the prediction horizon.
One of the controller's objectives here is to minimize the deviation on $[L_1\ L_2\ F_4]^T$, thus a large weight on particular output causes the controller to minimize deviations in that output. In this case we want to minimize deviation in the output flow F_4, with that a large weight value on F_4 will be a natural choice. See table 3.6.

Table 3.6 Weight on Controlled variables

Variables	Weights MPC2	Weights MPC3
F_4	10	75
L_1	0.031	0.035
L_2	0.031	0.035

For further reading on tuning parameters within MPC toolbox see appendix A.

Constraints

Constraints node within the MPC toolbox allows us to set up constraints on manipulated and output variables. Constraints can be hard or soft; after creating the MPC controller from Simulink model all constraints are unconstrained by default.

Before specifying constraints, one should know that manipulated variables constraints are hard by default whereas controlled variable constraints are soft. This setting can be changed by using the constraints softening button. The constraints considered in this project are stated in table 3.7.

Table 3.7 constraints on manipulated and controlled variables

Variables	constraints
F0	$0 \le F_0 \le 1$
SptL1	$0 \le SptL_1 \le 120$
SptL2	$0 \le sptL_2 \le 72$
F_4	$0 \le F_4 \le 1$
L_1	$0 \le L_1 \le 120$
L_2	$0 \le L_2 \le 72$

4. Results

4.1. Simulation scenarios

In order to compare the different control structures studied in this project, one simulation scenario was run in all cases to see how they respond to possible disturbances. The scenario considered in this project was:
The feed flow into the plant was assumed to be 1 m^3/min; disturbances were chosen to be 50 % of the feed flow in all cases. Step change times of disturbances into the process are shown in figure 4.1.

Table 4.1 disturbances step change

Disturbance	Step change time [min]
d_1	300
d_2	600
d_3	1000
d_4	1500

4.2. Single loop control

In this control structure the production rate was controlled by manipulating the feed flow F_0. Using a single loop-PI control on the TPM as shown in figure 2.2 the controller receives the signal from the output downstream F_4 and calculate the set-point of F_4 giving the required input flow F_0.
In addition there were two level controllers, one on the outlet flow of unit one and the second on the outlet flow of unit two. The controller used for the level control was proportional controller.
SIMC tuning rules was used to calculate the tuning parameters for both P and PI controllers; the closed loop time constant τ_c was chosen to $\tau_c = \theta$ for a tight control and $\tau_c = 3\theta$ for smooth control of F_4 sees table 3.1 and table 3.2.
The simulations results obtained are shown as plots of controlled variables, the responses obtained using $\tau_c = \theta$ are illustrated by "blue curves" and $\tau_c = 3\theta$ responses by "green curves". From figure 4.2 we observe that $\tau_c = 3\theta$ respond to disturbance slower than $\tau_c = \theta$, and that the back off on F_4 was 3,4 % higher when using $\tau_c = 3\theta$. The variations in hold up volumes are shown in figure 4.3 and 4.4.
The back off obtained using this control scheme was calculated using equation (2): 0.4227 and 0.4371 m^3/min successively for tuning with $\tau_c = \theta$ and $\tau_c = 3\theta$.

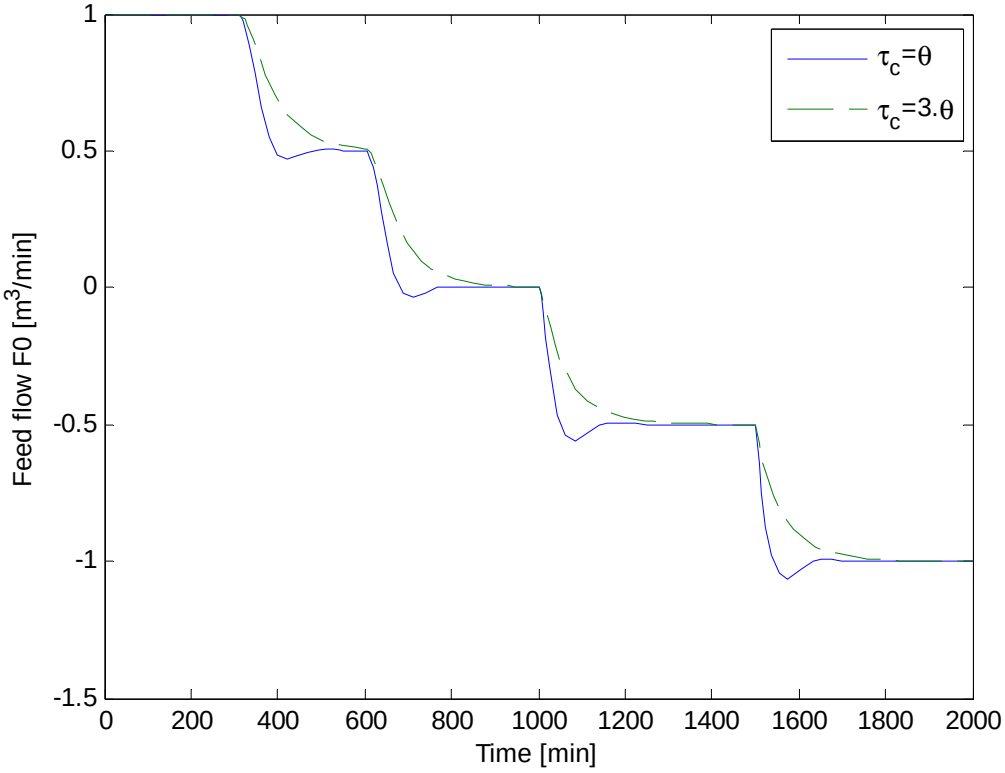

Figure 4.1 Single loop control response for F_0 with tuning $\tau_c = \theta$ (solid) and $\tau_c = 3\theta$ (dashed)

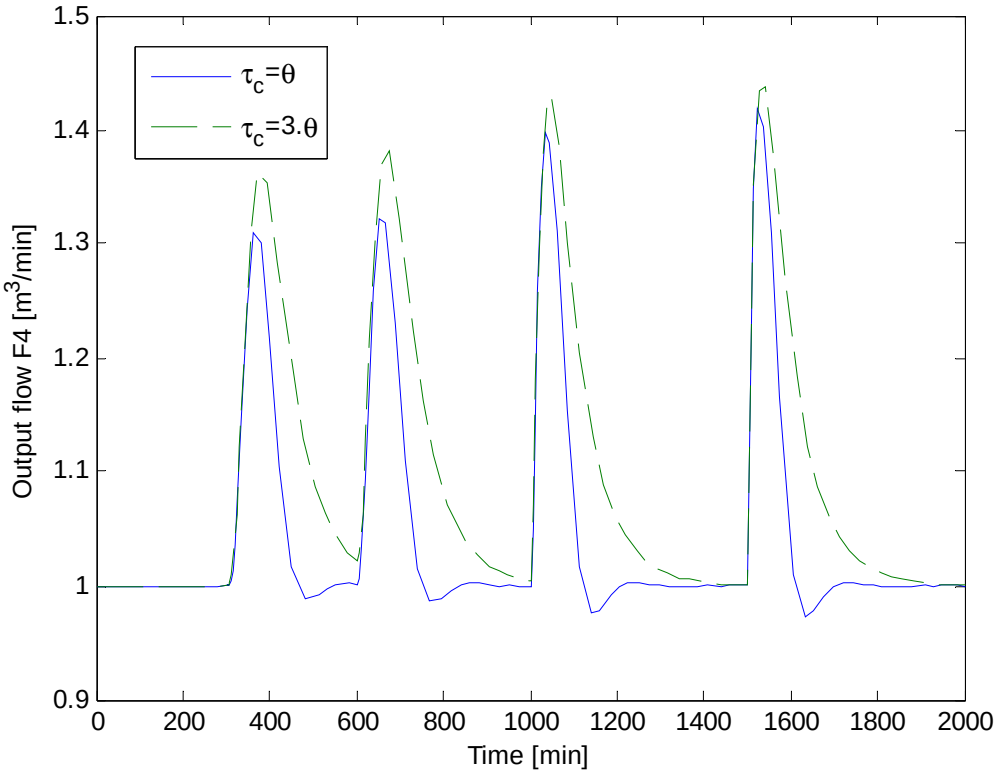

Figure 4.2 Single loop control response for F_4 with tuning $\tau_c = \theta$ (solid) and $\tau_c = 3\theta$ (dashed)

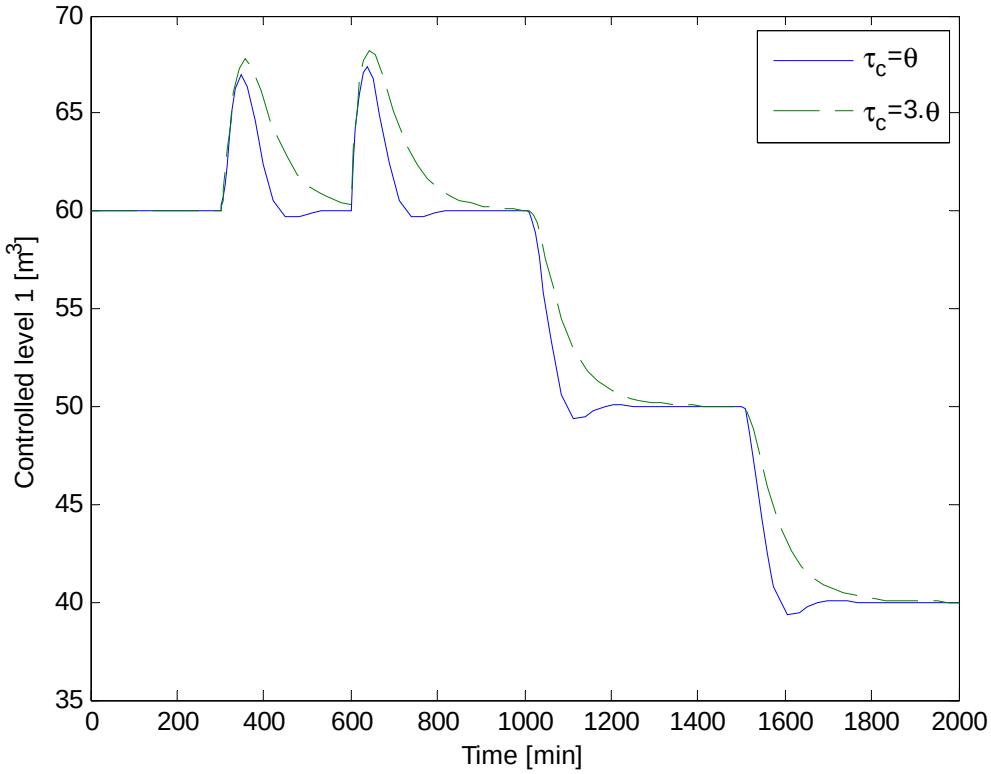

Figure 4.3 Single loop control response for level 1 with tuning $\tau_c = \theta$ (solid) and $\tau_c = 3\theta$ (dashed)

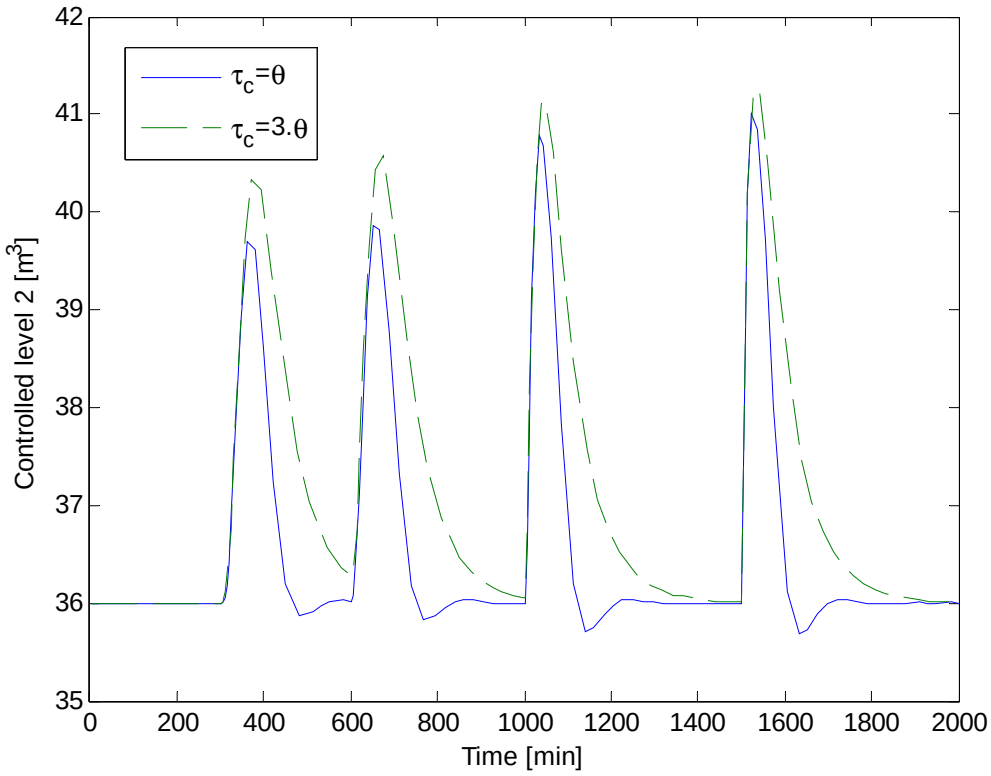

Figure 4.4 Single loop control response for level 2 with tuning $\tau_c = \theta$ (solid) and $\tau_c = 3\theta$ (dashed)

4.3. Single loop-feedforward control

In order to reduce the long loop (thus reduce the back off) encountered using single loop control and optimally exploit the buffer volumes in the process, a feedforward (ration) control was combined to the single loop. The tuning parameters used for the single loop-feedforward control scheme are stated in table 4.2. Figures 4.5-4.8 shows the results obtained using the actually control scheme.

Figure 4.5 shows the responses of the output controlled variable F_4, one using proportional controller to control levels in the process and the second using PI controllers on both levels in the system. Using single P controllers on level do not bring those at their set-point, whereas a little integral action (PI) brings the volume levels at their set-point (figure 4.7-4.8). However using PI controllers on the levels can results in larger back off on F_4. See figure 4.5. The Back off on F_4 using this control structure was calculated to 0.0769 and 0.0810 m^3/min for P and PI controller.

Table 4.2 Tuning parameters for the single loop-feedforward control

Parameters	K_c	τ_I
TPM	0.2	0.5
Level 1	0.05	100
Level 2	0.08	100

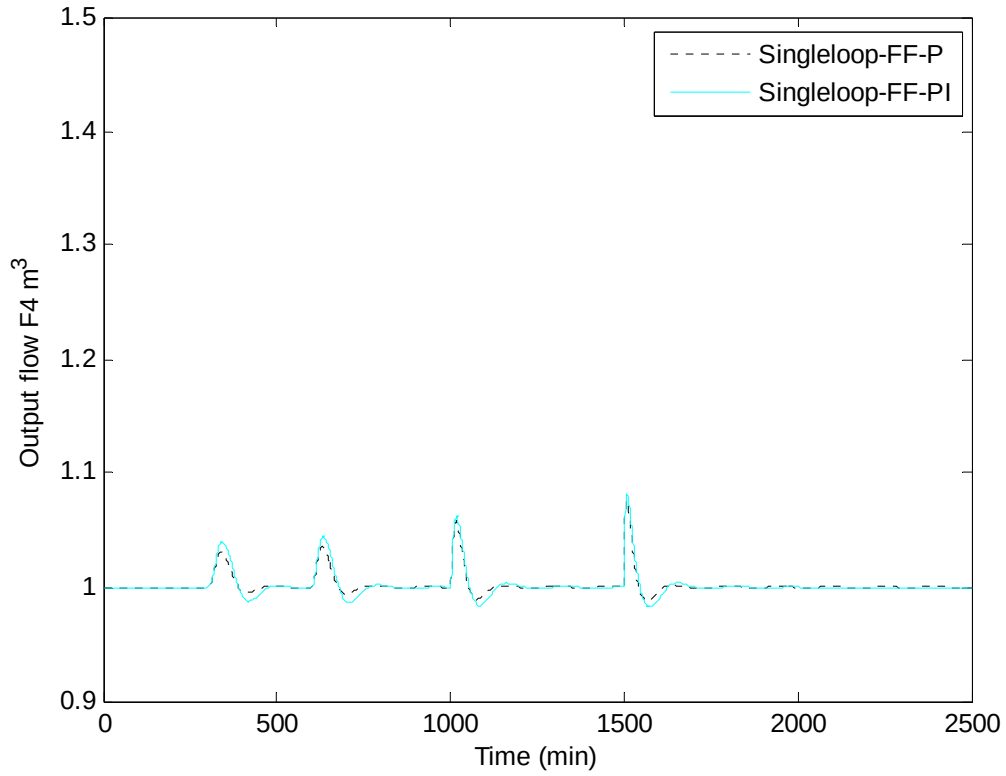

Figure 4.5 Single loop with feedforward control response for F_4 with P controller (dashed) and PI controller (solid)

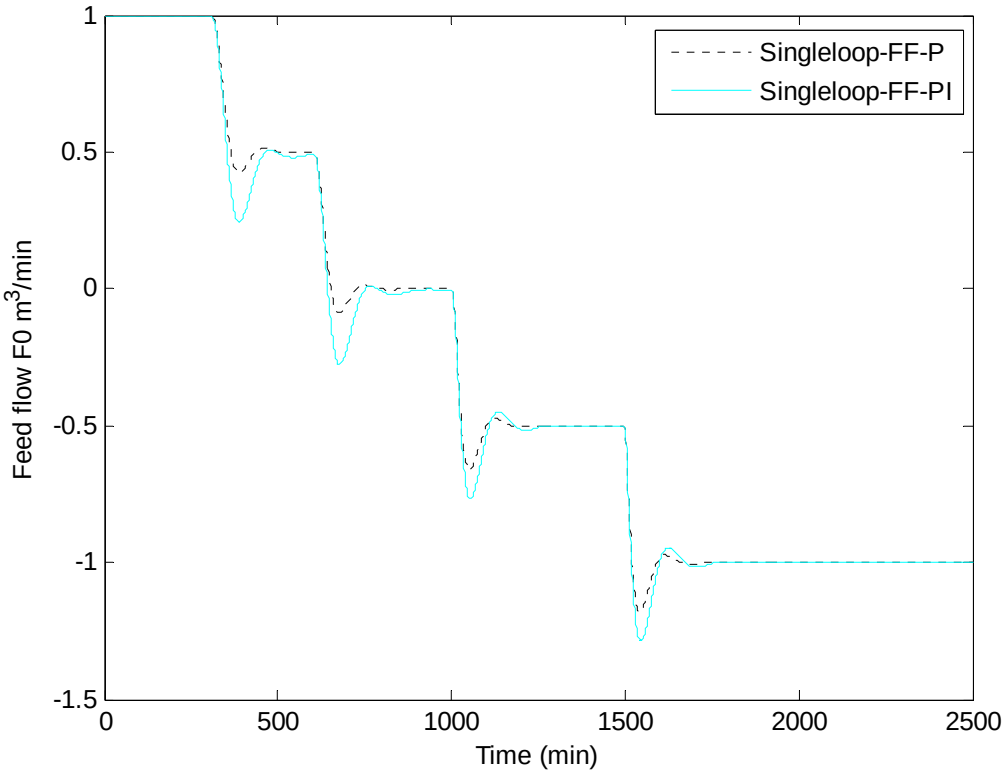

Figure 4.6 Single loop with feedforward control response for F_0 with P controller (dashed) and PI controller (solid)

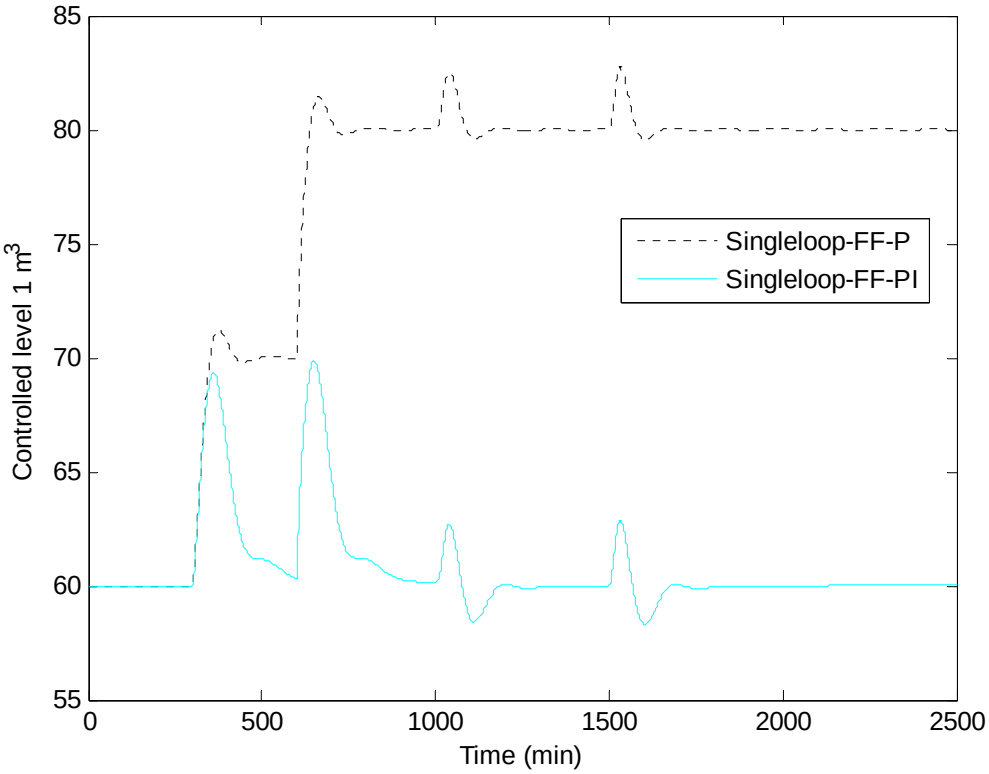

Figure 4.7 Single loop with feedforward control response for level 1 with P controller (dashed) and PI controller (solid)

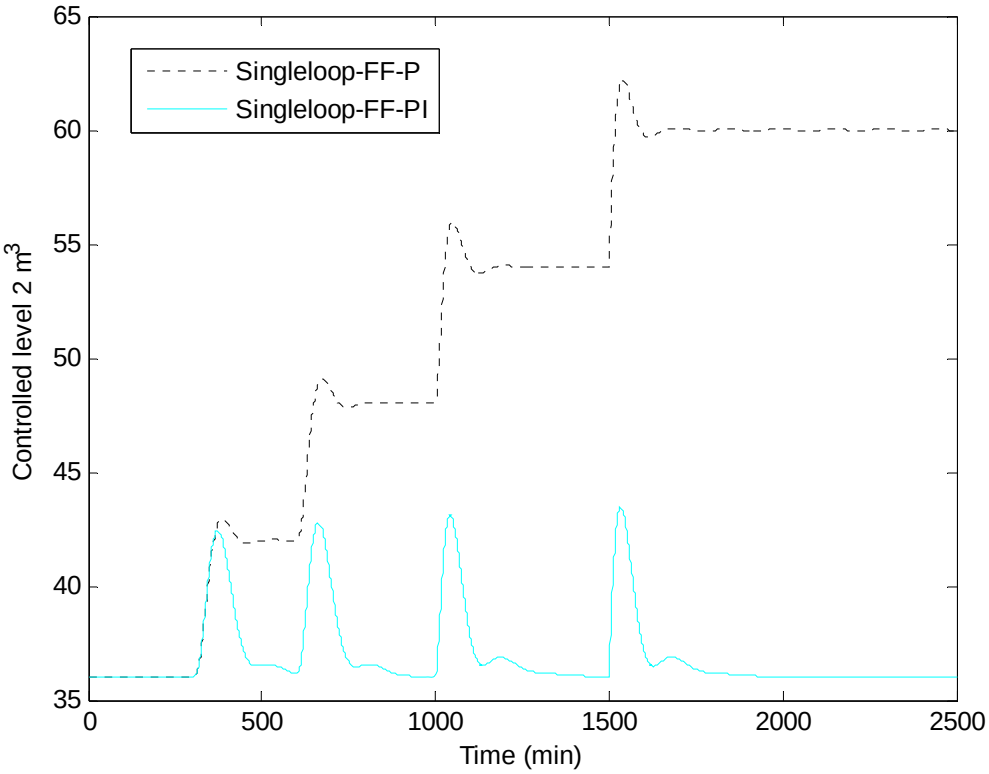

Figure 4.8 Single loop with feedforward control response for level 2 with P controller (dashed) and PI controller (solid)

4.4. Cascade control

By introducing extra measurements, we intended to intercept the effect of disturbances before they reach the output flow F_4, thus minimize the back off on F_4. The tuning parameters used for the PI-controllers in figure 3.4 are stated in table 3.3.
First, we measured variables F_2 and F_4, where F_2 is the secondary variable and controlled by the fast inner loop (TPM2), F_4 is the primary variable and controlled by the slower outer loop (TPM1). From figure 3.4 there are two disturbances (d_1 and d_2) that are handled by the extra measurement F_2 flow. Results from this implementation are shown in figure 4.9-4.12.
Figure 4.10 shows that measuring F_2 as an extra measurement to F_4 reduces considerably the effect of d_1 and d_2 on F_4. On the other hand d_3 and d_4, that enter the process after F_2, are unchanged.
Thereafter we relocated the extra measurement on F_3, in order to simultaneously handle d_1, d_2 and d_3. The result from this measurement shows that d_3 was reduced but d_1 and d_2 were poorly handled compared to F_2 measurement. From this observation a third implementation was made simultaneously on F_2 and F_3. See appendix B for the plots of the two later named measurements. Table 4.3 shows the back off resulted from the cascade control scheme.

Table 4.3 Back off on F4 using cascade control

Extra measurement	Back off [m^3/min]
F2 with $\tau_c = \theta$	0.4347
F2 with $\tau_c = 3\theta$	0.4568
F3 with $\tau_c = \theta$	0.3966
F3 with $\tau_c = 3\theta$	0.4352
F2-F3 with $\tau_c = \theta$	0.4126
F2-F3 with $\tau_c = 3\theta$	0.4523

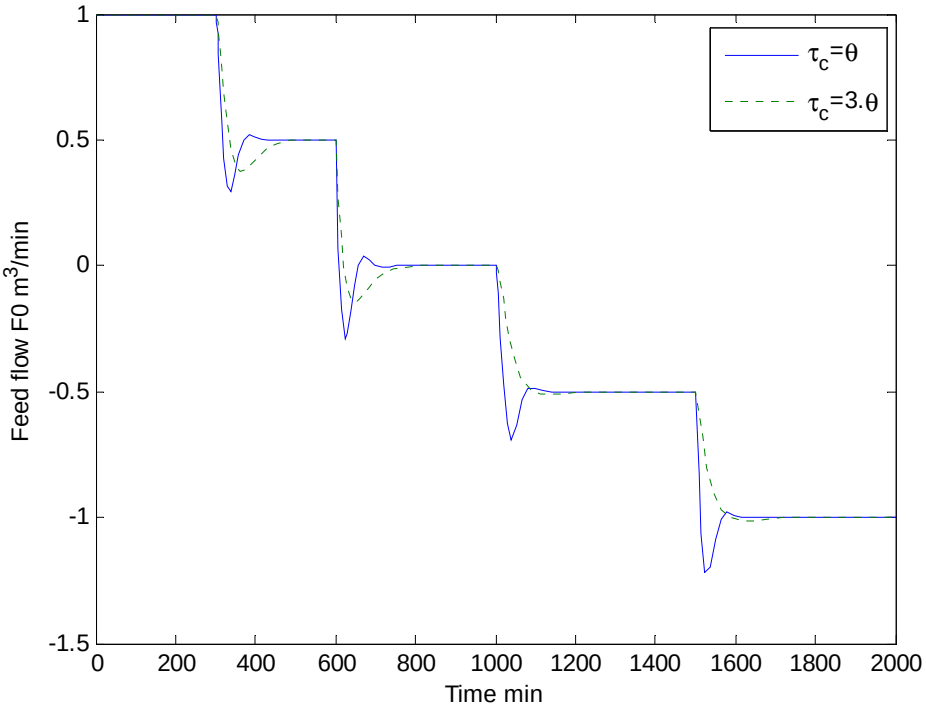

Figure 4.9 Cascade control response of F_0 using F_2 extra measurement with tuning $\tau_c = \theta$ (dashed) and $\tau_c = 3\theta$ (solid).

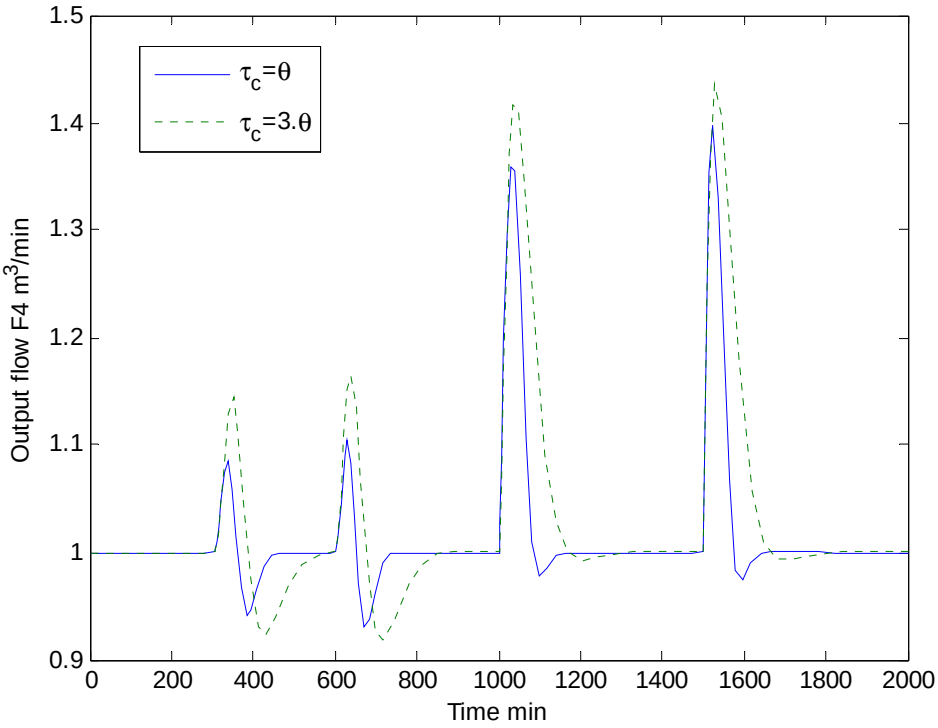

Figure 4.10 Cascade control response of F_4 using F_2 extra measurement with tuning $\tau_c = \theta$ (dashed) and $\tau_c = 3\theta$ (solid).

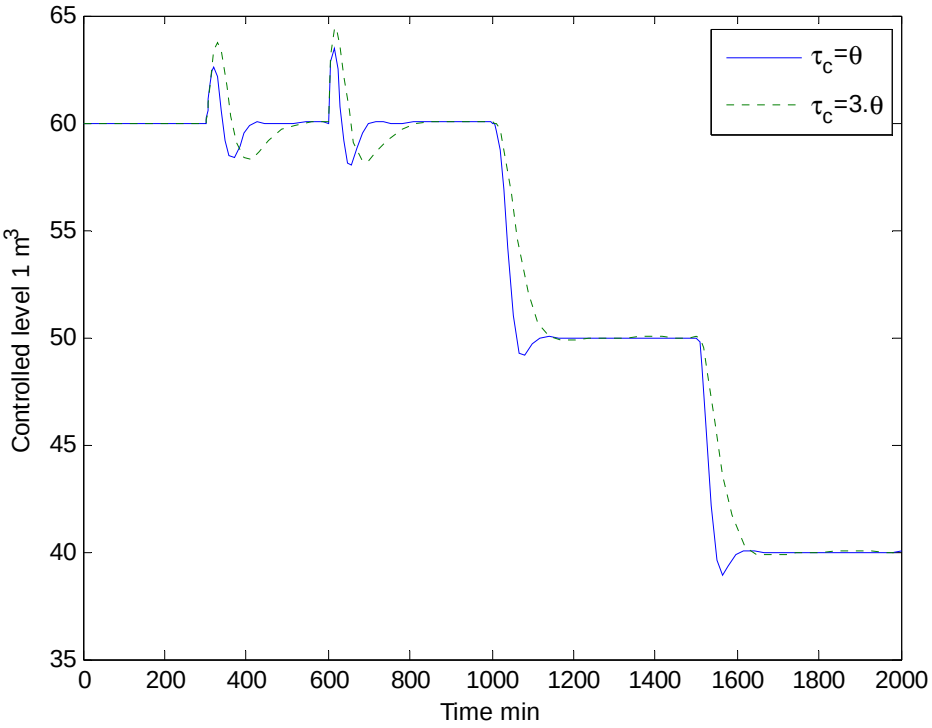

Figure 4.11 Cascade control response of level 1 using F_2 extra measurement with tuning $\tau_c = \theta$ (dashed) and $\tau_c = 3\theta$ (solid).

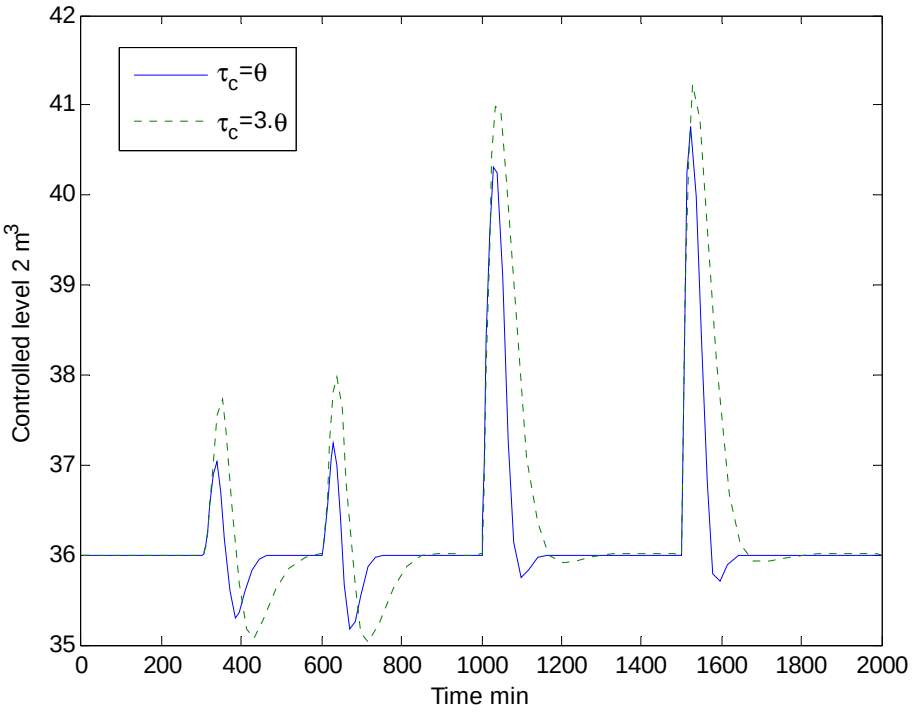

Figure 4.12 Cascade control response of level 1 using F_2 extra measurement with tuning $\tau_c = \theta$ (dashed) and $\tau_c = 3\theta$ (solid).

4.5. MPC controller

For model predictive control, two different MPC controllers were studied and applied on the Simulink model in figure 3.5. Tables 3.4 – 3.6 (section 3.4.2), show the different tuning parameters used to tune those controllers. All manipulated variables [F0 SptL1 SptL2] ^{T} and controlled variables [F_4 L_1 L_2] ^{T} were assigned constraints, see table 3.7.
Figure 4.13 – 4.19 show the responses obtained using model predictive control scheme. MPC1 controllers has large weight rate value on the manipulated variable F_0 (table 3.5). Which penalizes large changes in this variable (figure 4.17) and the lower weights rates on levels allow those to vary quite freely (respecting constraints see figure 4.18-4.19). F_4 response obtained from this controller was aggressive and varied a lot from its set-point when disturbances occurred, see figure 4.13.
On the other hand using MPC2 controller, large weight value on the output controlled variable F_4 resulted in tight bottleneck control on this variable (figure 4.13). However this implementation caused large variation on the manipulated variable $F_{0;}$ see figure 4.17.
Using equation (2) the back off in this control scheme was calculated to 0.0552 and 0.0509 m^3/min successively for MPC1 and MPC2 controller.
All results are presented as plots of controlled variable, set-points and manipulated variables.

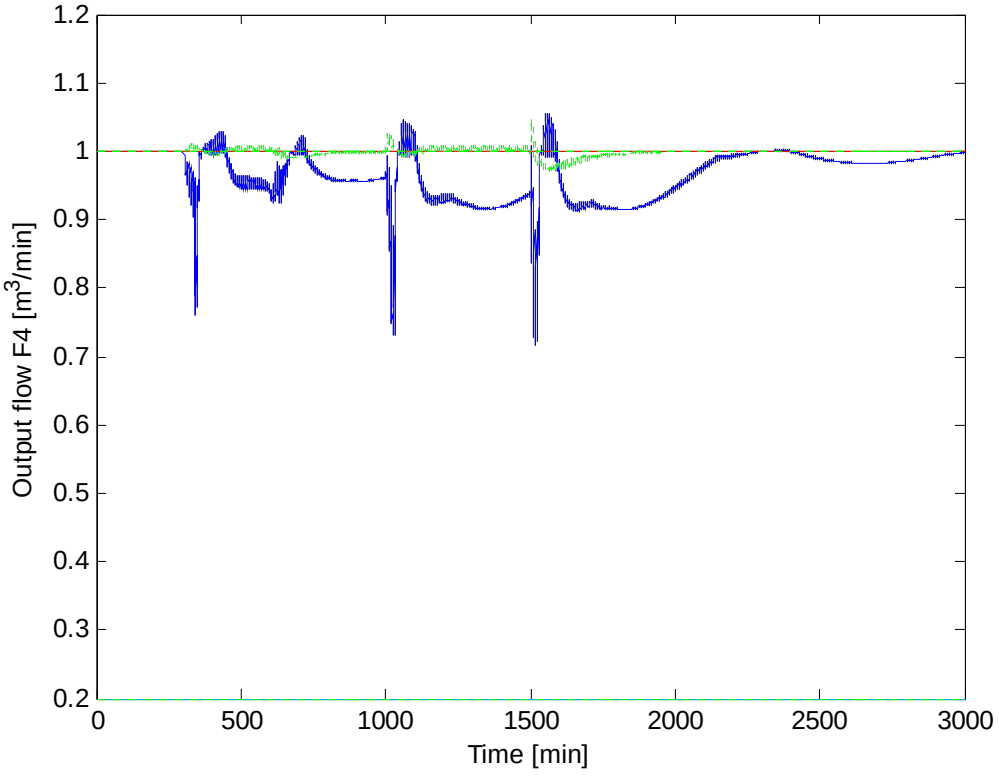

Figure 4.13 MPC control response for F_4 with MPC1 (solid) and MPC2 (dashed)

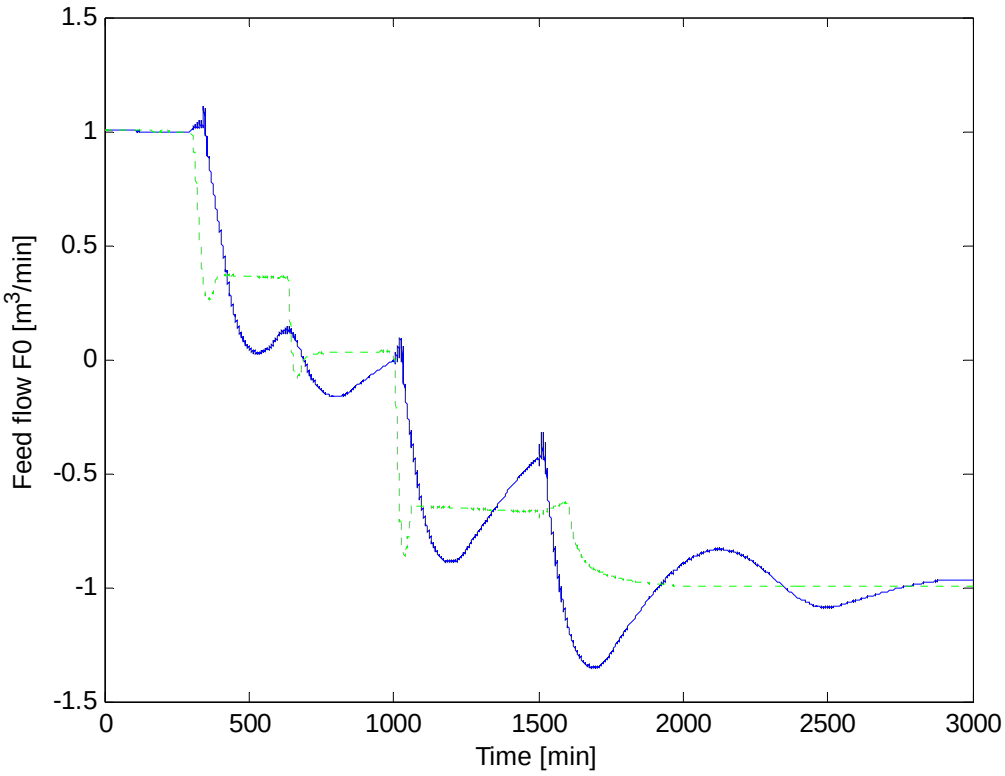

Figure 4.14 MPC control response for F_0 (into the process model out of TMP (PI)) with MPC1 (solid) and MPC2 (dashed)

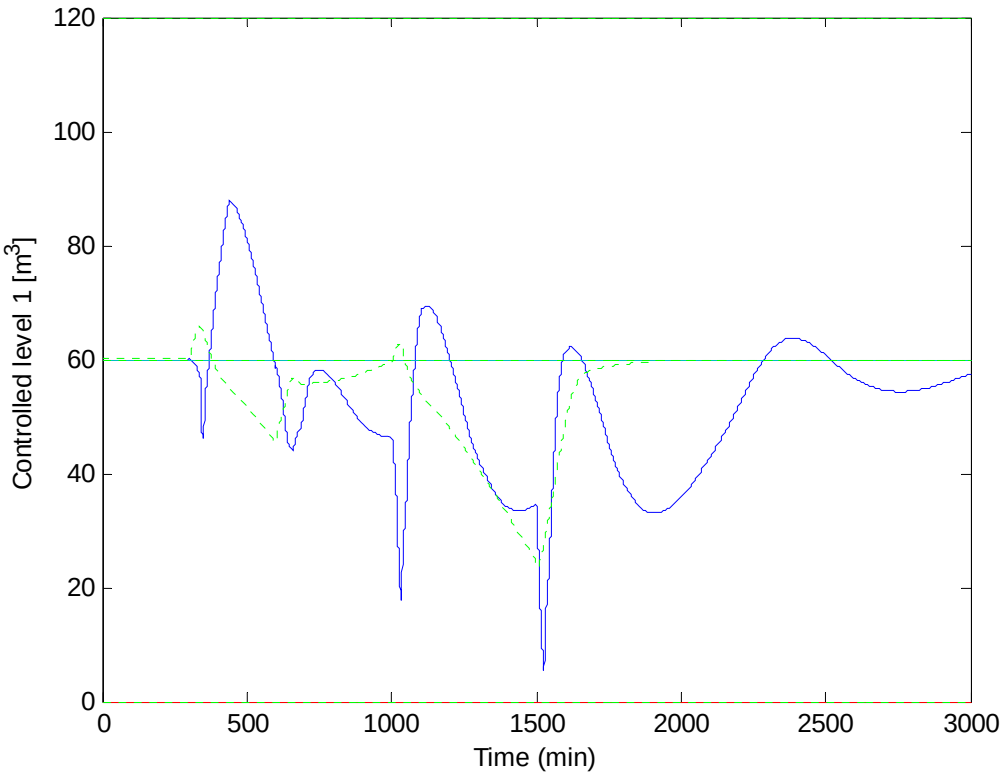

Figure 4.15 MPC control response for L1 with MPC1 (solid) and MPC2 (dashed)

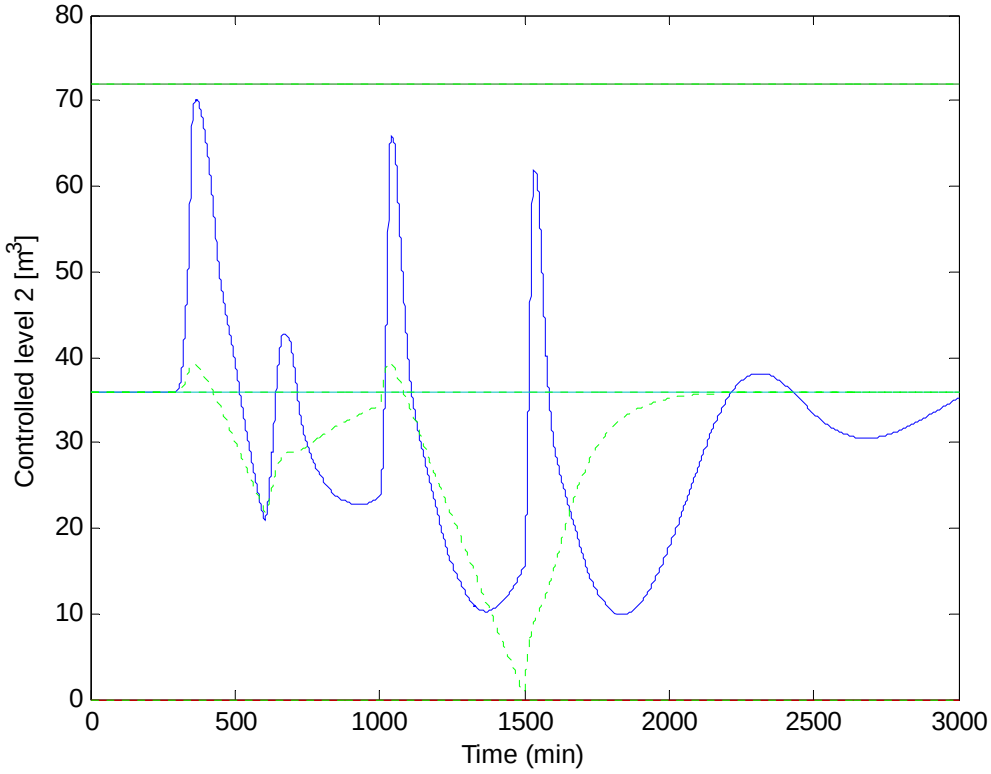

Figure 4.16 MPC control response for L2 with MPC1 (solid) and MPC2 (dashed)

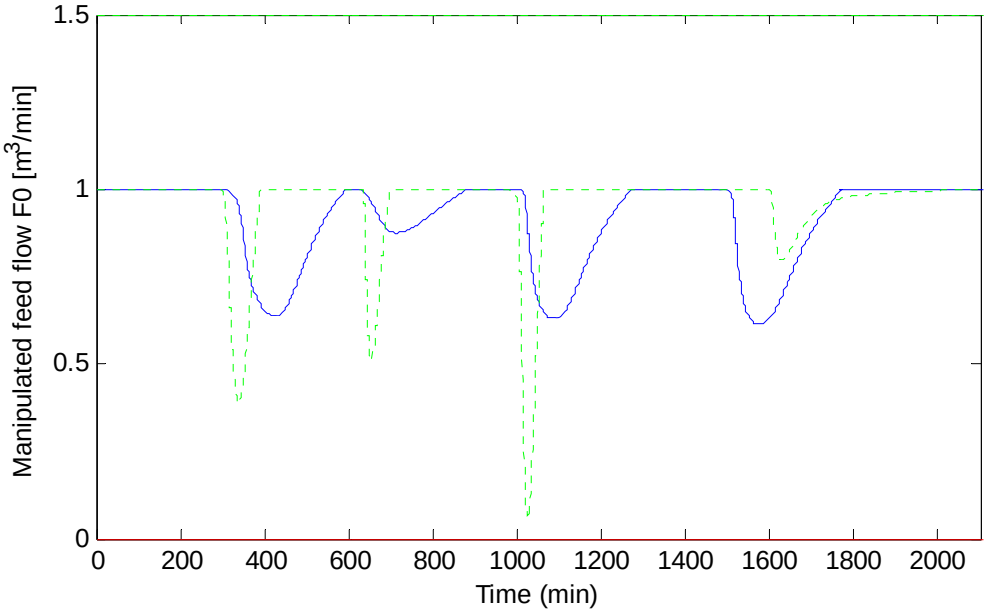

Figure 4.17 MPC control response for manipulated F_0 with MPC1 (solid) and MPC2 (dashed)

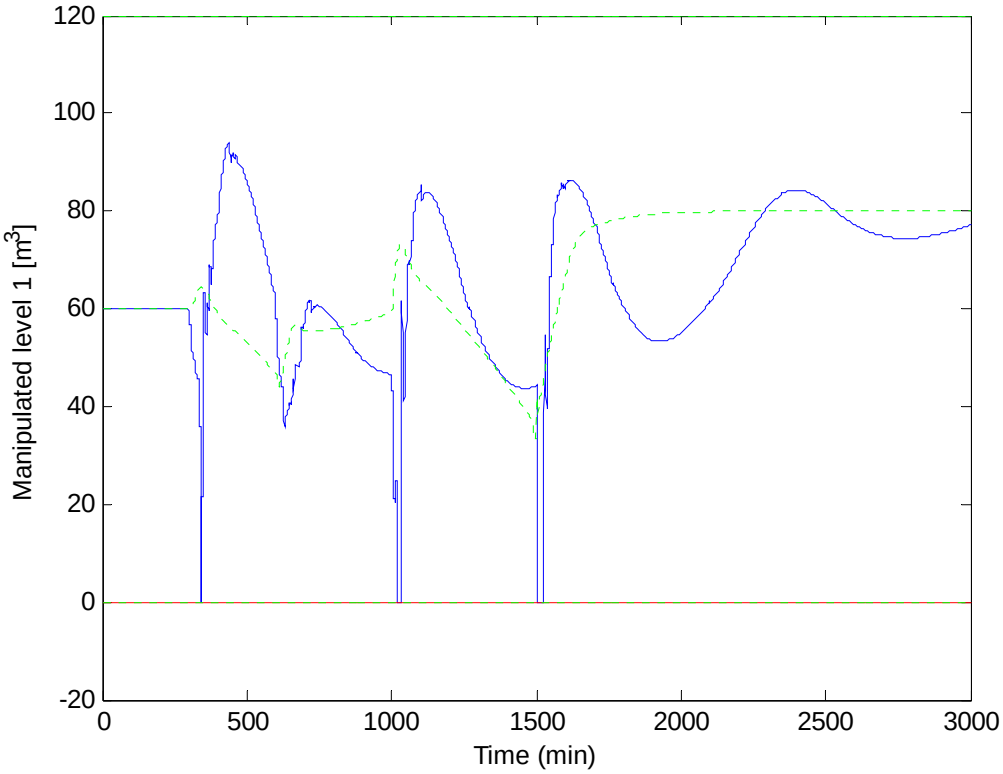

Figure 4.18 MPC control response for sptL1 with MPC1 (solid) and MPC2 (dashed)

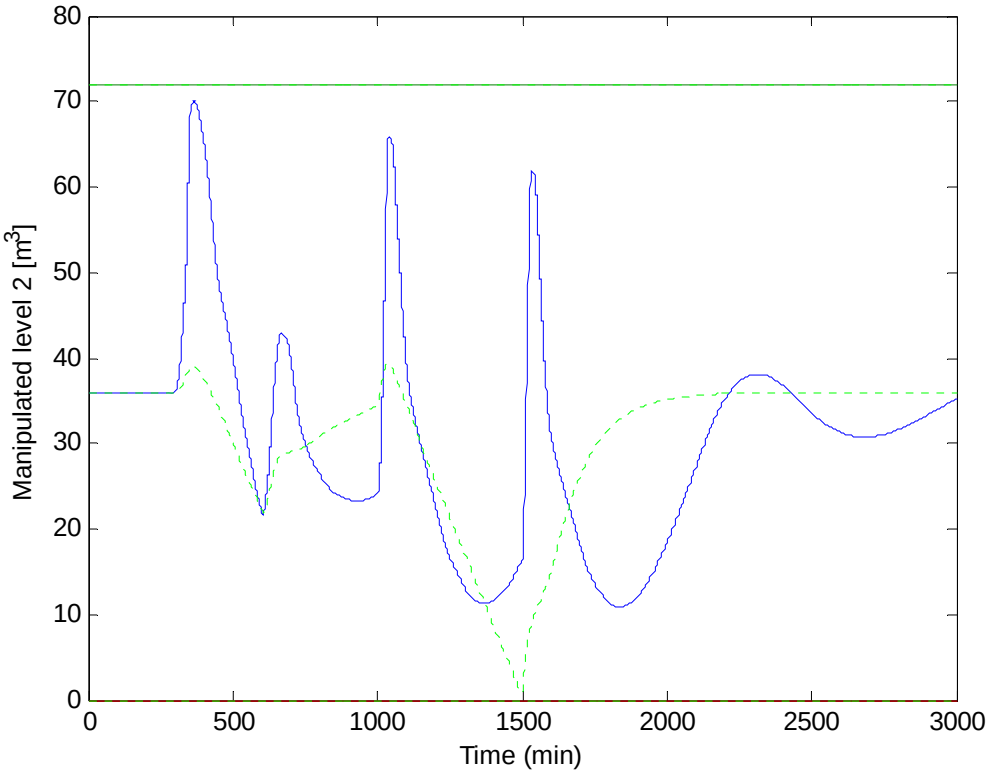

Figure 4.19 MPC control response for sptL2 with MPC1 (solid) and MPC2 (dashed)

4.6. Comparison of results

In section 4.5 we present the results obtained in this project in details for each control scheme. Here we are going to compare the important results obtained from those control structures. Figures 4.20 shows the response of F_4 obtained using single loop control with $\tau_c = \theta$ as a choice for the closed loop time constant τ_c, single loop with feedforward on level control with P as controller for levels, Cascade control with F_2 as secondary controlled variable and F_4 primary controlled variable, and model predictive control with MPC2 as controller.

Taking the single loop control structure as reference for comparison, table 4.4 shows how much the back off resulted from this control structure can be reduced (or increased, but not the goal here) using the remainder control structures. Range column (1 – 4; 1 being the best) in table 4.4 shows which control scheme gives minimum back off on F_4.

Table 4.4 Back off

Control structure	**Back off** (m^3/min)	**Reduction (%)**	**Range**
Single loop control	0.4227	- Reference -	4
Single loop with feedforward	0.0769	35.37	2
Cascade control	0.3966	2,610	3
Model predictive control	0.0509	37,20	1

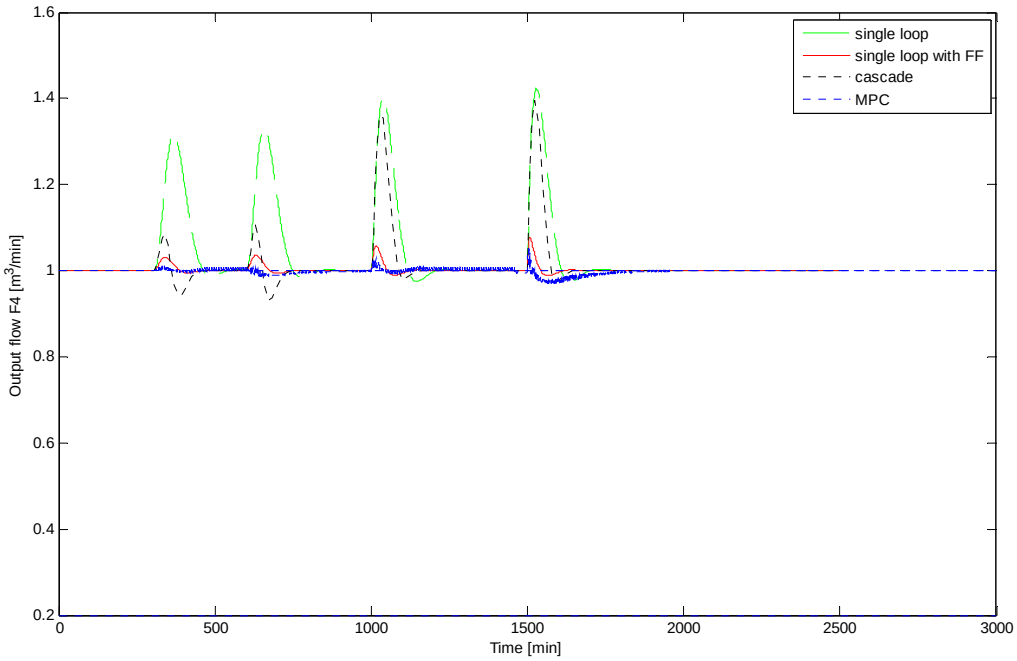

Figure 4.20 Response of output flow F_4 obtained using single loop, single loop with feedforward, cascade and MPC control structure

5. Discussion

5.1. Single loop control

Here we assumed the simplest case, where the bottleneck is fixed. The considered process model in this project had the TPM on the feed rate and buffer volumes placed upstream the bottleneck.
A single loop-PI controller was used on the TPM and tuned by using Skogestad's tuning rules (SIMC). The closed loop time control τ_c was chosen to be $\tau_c = \theta$ for robustness and tightly control of the bottleneck, while for a smoother bottleneck control, it was chosen to be $\tau_c = 3\theta$.
The first named choice resulted in fast response and well set-point tracking compare to the latter choice. In this case where we are interested in tight bottleneck control and minimum back off; $\tau_c = \theta$ is to be preferred.
This control scheme gave poor response on the disturbances into the process, which results in quite large back off (43 %). The reason is that hold-up volumes between the TPM and the bottleneck increase the effective delay and the tight control of the bottleneck becomes more difficult.
However this long loop could be reduced by replacing the TPM closer the bottleneck flow. But this requires a permanent reassignment of the levels loops. For moving bottleneck a better approach is to use a multivariable controller, where input and output constraints are included directly in the problem formulation (Aske et al., 2007).
In order to reduce the long loop encountered using single loop control, other control structures should be considered.

5.2. Single loop with feedforward control

In this project we used a special case of feedforward called ration control; the signal from the TPM is added to the level controllers. By measuring disturbances and taking corrective action before the controlled variable is upset, single loop-feedforward control provided dramatic improvements for regulatory control in this project. The back resulted on F_4 using this control scheme was 7 % of the feed rate. Compare to single loop control by itself, the back off on F_4 was reduced by 35 % using single loop-feedforward control. The reason is that effective time delay its reduced, which facilitate tight bottleneck control and buffers volumes are optimally exploited compare to single loop control alone. However the chief disadvantage of feedforward control is that the disturbances variables must be measured or estimated on line, which is not always possible (Seborg et al., 2004).

5.3. Cascade control

Extra measurements were introduced in attempt to reduce the effective delay encountered using single loop, thus facilitate the tight bottleneck control. A secondary loop was implemented on the secondary flow F_2; the inner loop effective delay was 5 minutes and the outer loop 16 minutes. The inner loop effective delay was clearly small compared to 24 minutes for the single loop. The result was that disturbances upstream the secondary flow

F_2 were well rejected and their effect on the primary controlled variable minimized compared to disturbances downstream the inner loop.
Moreover, we introduced a new extra measurement on F_3. The idea was to intercept disturbances d_1, d_2 and d_3 simultaneously, this was achieved but the response on d_1 and d_2 was not as good as the response obtained from the F_2 measurement. The reason is the increased effective delay from 5 to 13 minutes for the F_2 and F_3 measurement. Finally we implemented two secondary loops simultaneously on F_2 and F_3, which means three PI-controllers in the control scheme instead of two PI-controllers used previously. The results for the last named measurement were not far from the results obtained using F_3 extra measurements. The effect of disturbances downstream the studied extra measurements (in this case d_4), on the primary controlled variable F_4 remained nearly unchanged compare to single loop – PI controller. The reason is that in generally the objective of the regulatory layer is to locally control secondary measurements so that the effect of the disturbances on the primary measurements can be handled by the above layer (Skogestad, 2004). Furthermore disturbances upstream the extra measurements have to be of big importance than those downstream the extra measurements. In this project all four disturbances were of same importance. Even if extra measurements reduced considerably back off caused by disturbances d_1 d_2 and d_3. Using equation (2) the back off resulted from cascade control was not far from the back off obtained using single loop control; in same cases it was even worst. As mentioned above the reason was that d_4 could not be well handled by this control scheme.
However a discussion with professor Skogestad about the process model considered in this project raised a question on the feasibility of the implementation of this control structure in practice.

5.4. Model predictive Control

Most problems encountered above were overcome using model predictive control. We used buffer volumes as degrees of freedom by adding level set-points as manipulated variables and formulated an MPC controller within the Simulink/Matlab inbuilt MPC controller block. The main objective here was to use buffer volumes dynamically to reduce the long loop in the process model. And thereby quickly dump disturbances before they affect the downstream flow F_4. The first task was to learn how the Simulink inbuilt MPC controller block works, Matlab software provides a clear and user friendly tutorial on this topic.
With its advantages of automatically tracking the moving constraints and reassigning control task in an optimal manner, MPC was suitable for an improved tight bottleneck control in the process model considered here.
The results obtained using model predictive controls were considerably better than those obtained using single loop and cascade control. The back off obtained using this control structures was 5 % of the feedrate, which means 38 % better than single loop and cascade control and 2 % better than single loop-feedforward control.
However it is more complex, and after its implementation it gives little access to the user, its failures and sensitivity to errors are almost unpredictable (Skogestad, 2004). That's why MPC is often placed on the top of the regulatory layer; and if it starts to misbehave, it is usually possible to disable it, and let the local loop controllers hold the plant at the last set-points they received from a higher layer. This was implemented in this project by enabling the input port for externally manipulated variables to the plant in the MPC controller block (see appendix A).

Another challenge encountered in this project, when using model predictive control was the tuning parameters. The reason is that there are many adjustable parameters in the predictive control (section 3.3.2) that affect the plant behaviours and those are mostly based on 'rule of thumb' gained from experience (Maciejowski, 2002).
To illustrate this we can take an example of the input and output weight tuning parameters:
Depending on the weight value assigned to the manipulated variable F_0 and the controlled variable F_4, a smooth control of the feed rate F_0 was observed using a larger weight value on this variable, while the control of F_4 was quite aggressive and showed a quite bad set-point tracking. On other hand a large weight value on the controlled variable F_4 gave smaller back off and tight tuning of this variable compared to the previous case of large weight value on F_0. However this resulted in more aggressive tuning of F_0 and poor set-point tracking.
So, the question is, do we accept an aggressive feed rate control to achieve smoother F_4 control or the inverse. One should try to find some parameters that give a kind of equilibrium between these variables.

6. Conclusion

In this project we investigated the possibility of maximizing plant throughput using four different control structures. The studied control structures were a single loop control, cascade control and model predictive control.
The single loop control structure, which is usually used in regulatory layer as the plant stabilizer, performed well on the process model studied in this project. However, it showed a significant weakness concerning disturbances rejections due to the long loop in the process. Significant improvement was achieved by adding feedforward control on level controllers. But feedforward control needs that the disturbances be measured or estimated on-line; which is not always possible.
In the cascade control structure, the extra measurements on F_2 and F_3 reduced considerably the local disturbances, thus minimized the back off resulted on those. Disturbances which entered the process after the extra measurements (in this case d_4) were not optimal handled.
The model predictive control was superior to the previous control structures. The back off here was smaller compared to the single loop control, which results in maximum throughput. It should be noted that complexity and non trivial tuning parameters are the disadvantages of this control structure.

The overall conclusion is that the single loop control structure was suitable for the basic control of the process model. The cascade control structure was appropriate for a local disturbance rejection; whereas the disturbances on the primary controlled variables were well handled by the upper layer, in this case supervisory layer with MPC. On the other hand significant improvement on single loop control was achieved by adding feedforward control on level controllers. Single loop with feedforward control scheme performed nearly as the model predictive control.
It should be noted that for level control, proportional controller only doesn't bring levels at their set-points. To achieve this one need an integral action (PI) (Skogestad, 2004). However, there is no point whatever in controlling the level in a buffer tank to a set-point, since that would destroy its purpose. The use of PI controller in single loop with feedforward control structure (section 4.3) confirms this affirmation.

Literature

Aske, E.M.B., Skogestad, S., & Strand, S. (2007). Throughput maximization by improved bottleneck control. In *8th International symposium on Dynamics and control of process systems (DYCOPS)*, vol. 1, Cancun, Mexico, pp.63-68.

Maciejowski, Jan Marian (2002). *Predictive control with constraints*. Printice Hall

Price, Randel M. and Philip R. Lyman and Christos Georgakis (1994). Throughpout manipulation in plantwide control structures. *Ind. Eng. Chem. Res.* **33**, 1197-1207

Seborg, Edgar, Mellichamp (2004) *Process Dynamics and Control, 2. ed.*, John Wiley & sons, Inc.

Skogestad, S. (2003). Simple analytic rules for model reduction and PID controller tuning. *J. Proc. Control* 13, 291–309.

Skogestad, S. (2004). Control structure design for complete chemical plants. *Computers and Chemical Engineering*, *28*, 219–234.

Skogestad, S., Postlethwaite, I. (2005). *Multivariable Feedback Control: Analysis and Design.* John Wiley & Sons.

Attached files

Single loop folder
Single loop with feedforward folder
Cascade control folder
Model predictive control folder
Comparison of results folder
See appendix C for more details on those folders.

Appendix A: MPC toolbox

Some of information presented in this appendix maybe found in the main report. The reason is that this appendix is written for those who are interested to know more about the MPC toolbox.

A.1 The MPC toolbox

If you want just to run the Simulink model (in figure 3.2) with MPC controller go direct to section A.2. Figure A.1 (A.0.1 this notation concern all figures in this appendix) shows the MPC controller block in Simulink library.

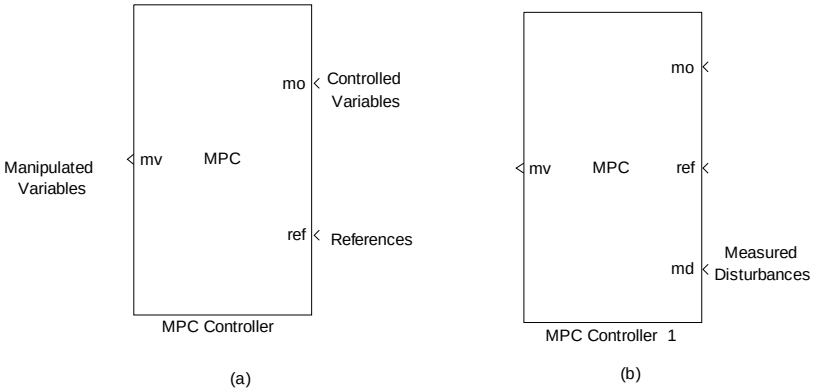

Figure A.0.1 click the design button from the controller mask to open the MPC design Tool and design the MPC controller object there.

A double click on the MPC controller in figure A.2 open the mask shows in figure A.3

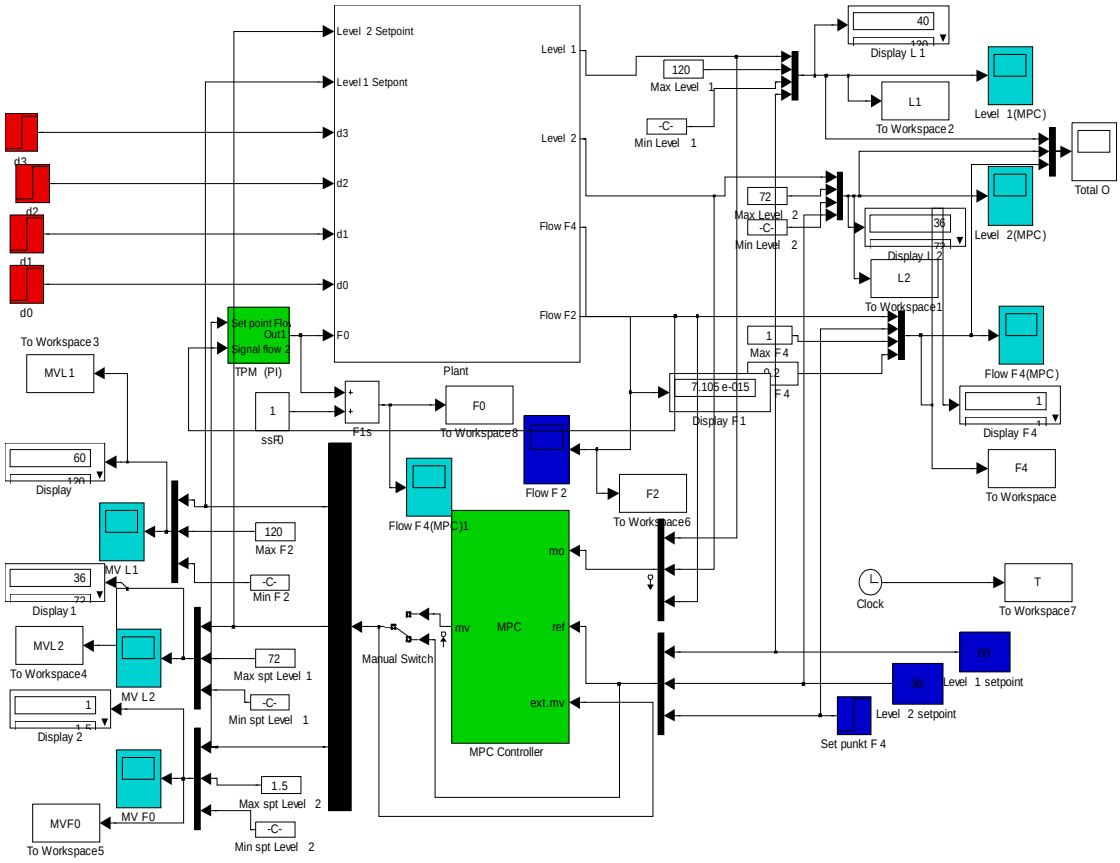

Figure A.0.2 The Simulink process model with MPC controller, the extern manipulated variable are enabled

The mask requires that we specify a valid MPC controller object. There are two ways of providing an MPC controller object:
- The first is to load an existing MPC object from the workspace.
- The second is to click the design button from the controller mask to open the MPC design Tool and design the MPC controller object there.
Here we used the second option
By clicking the Design… button in figure A.3 one has to specify how many manipulated variable are in the system, in this project we have got three MVs, which are the feed flow F_0 and the set-points of the two levels in our process model; $[F_0\ sptL_1\ sptL_2]^T$. After specifying the MVs one hit then the Ok button and the MPC tool design perform the operations presented in figure A.4. When these operations are finished, we have a linearized discrete-time; state model of the form presented in equation (A1) - (A2) and a default MPC controller has been constructed. The MPC controller is ready to be adjusted.

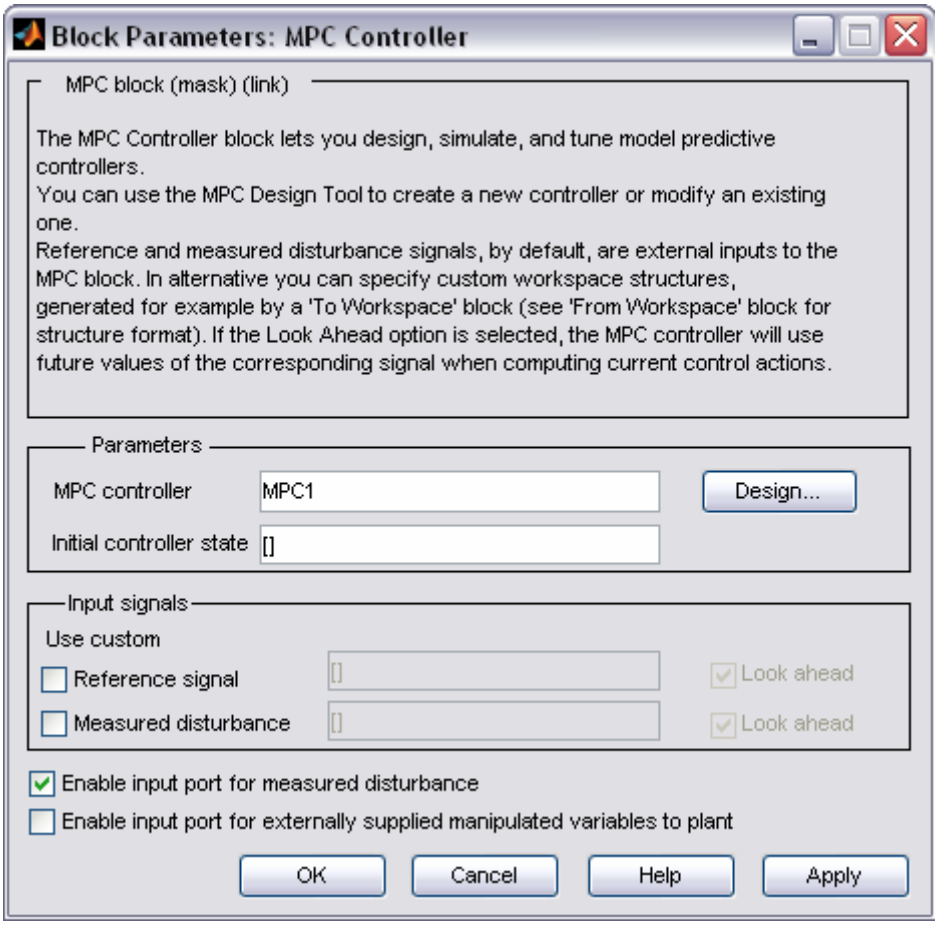

Figure A.0.3 MPC controller mask

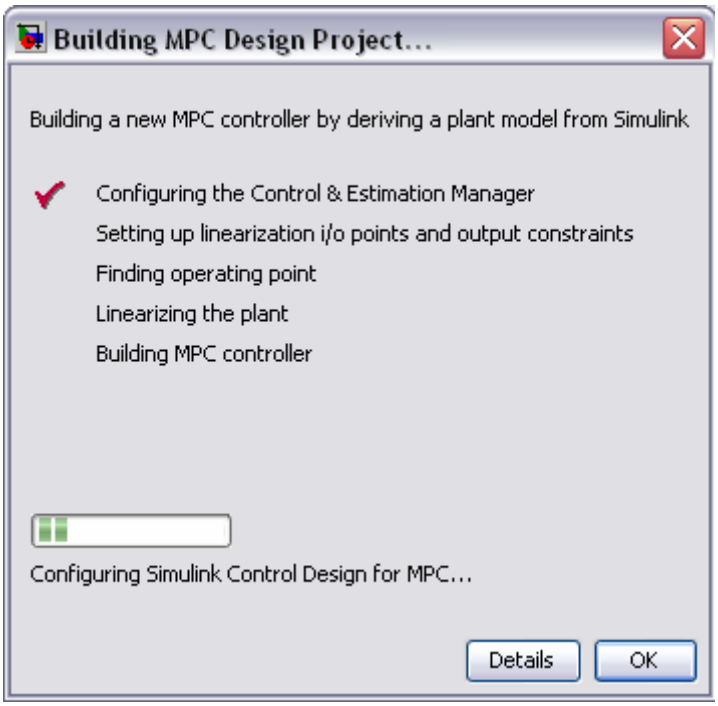

Figure A.0.4 building of MPC controllers for the Simulink Process model

$$x_{k+1} = Ax_k + Bu_k \quad \text{(A1)}$$

$$y_k = Cx_k \quad \text{(A2)}$$

Where x is the state vector, u the input vector; y is the controlled outputs vector.

Hitting the Ok button in figure A.4 gives us the possibility of tuning the default built MPC controller. The view shown in figure A.5 is the MPC structure overview; we access to the linearized model by clicking the plant models node and the default MPC controller by clicking the controller's node (from here we can tune the controller). The scenarios node gives us the possibility of testing different MPC controllers within MPC tool manager before we can choose one of them if satisfy.

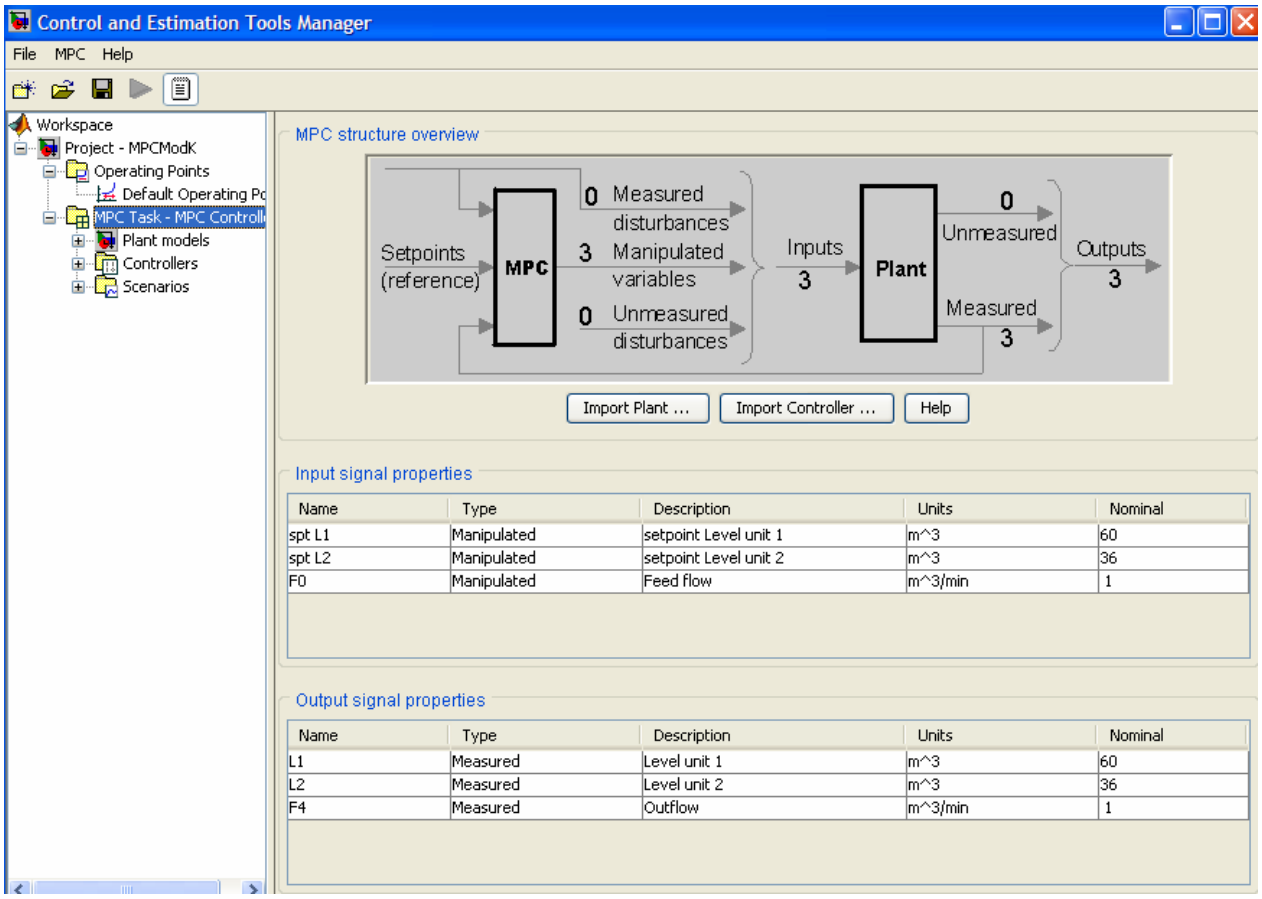

Figure A.0.5 Control and estimation tools manager window: Shows the MPC structure overview

We can run closed-loop simulations (the model in Simulink) while we are editing the MPC controller in the MPC design Tool (figure A.5). Before closing the MPC design tool one has to export the designed controller to the workspace or eventually save it as a Matlab file for later use. Then it can be loaded as an MPC object to the workspace.

When one needs to switch between MPC control and another type of control (e.g., manual control, PI-controller in this case), one can enable the extern manipulated variable port of the MPC controller block (see figure A.6)

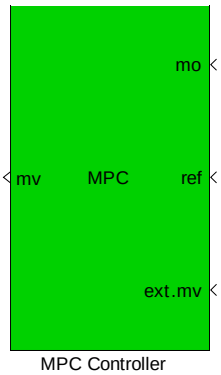

Figure A.0.6 MPC Simulink library with enabled external manipulated variables

Using the same approach one can enable or disable measured disturbances. If they are disabled the block has two input signals, namely measured outputs and references. When they are enabled, the block has measured disturbances as the third input signal (see figure 8.1).

A.2 MPC tuning

Prediction and control horizon

For prediction and control horizon tuning see the main report.

Weight tuning

Clicking on the controller's node from figure A.5 and selecting the weight tuning tab, the view in figure A.7 appears. This let us tune the output $Q(i)$, the input $R(i)$ weighting matrices and the overall slider control. The last named adjusts the weights on all variables simultaneously, a large value gives fast response whereas a lower value gives a more robust response. The overall slider control (in figure A.7) can be adjusted between 0 and 1.

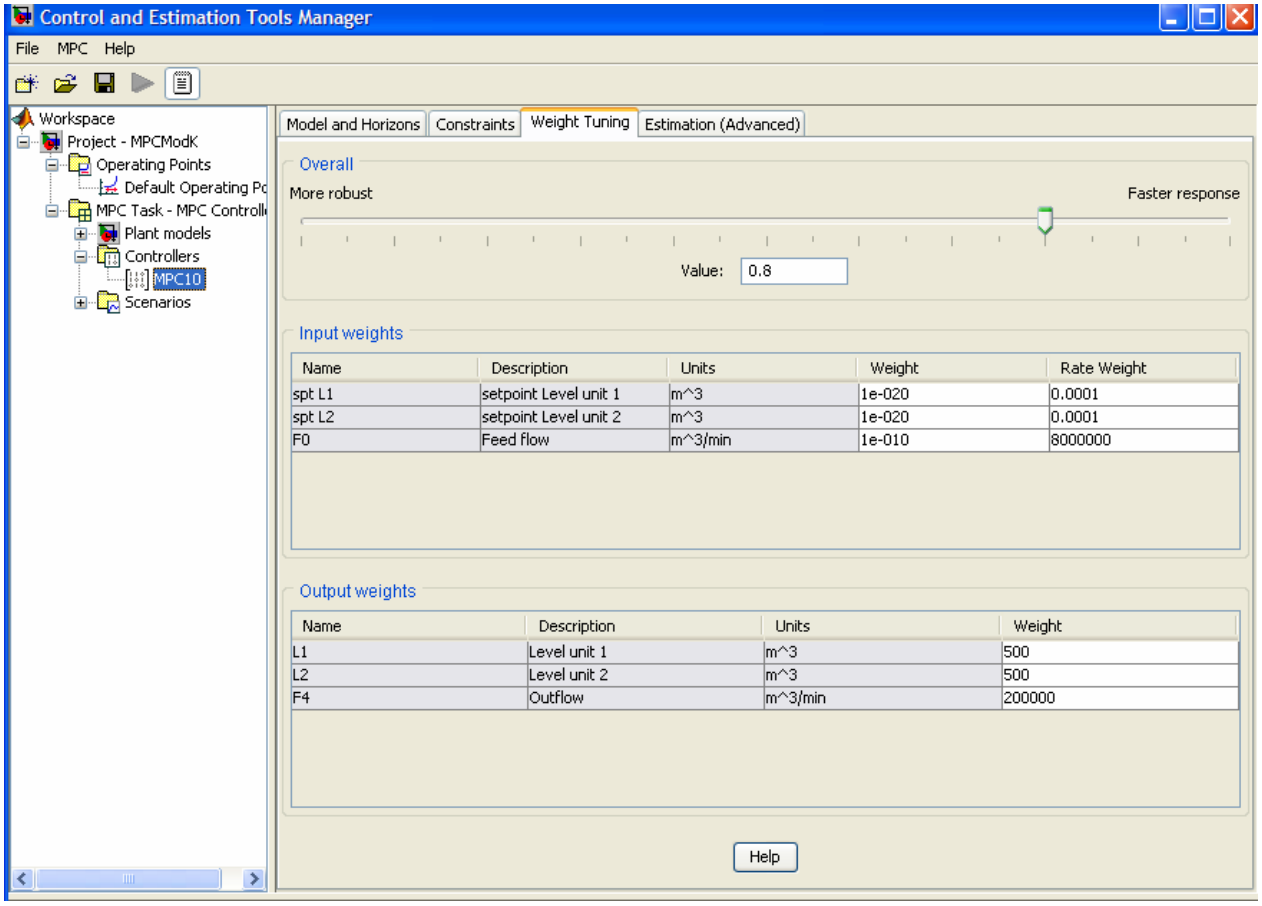

Figure A.7 Weight tuning tab

Inputs weight

For the input weight, we have got to tunings variables, weight and weight rate (see figure A.7). Name, description and units columns can not be edit form the weight tuning tab, to edit these we use the signal definition view shown in figure A.5.

The weight column penalizes deviations on each manipulated variable from its nominal value using $S_u(k)=\sum_{i=1}^{M}\sum_{j=1}^{n_{mv}}\left\{w_j^u\left[u_j(k+i-1)-\bar{u}_j\right]\right\}^2$ where w_j^u is the input weight, $\bar{u}_j$ is the nominal value for input j, and $S_u(k)$ is the weighted sum of squared deviations.

The default weight is zero, which means that the corresponding MV can vary freely provided that it satisfies its constraints. The chosen weight must be zero or a positive real number. A larger weight value keeps its corresponding manipulated variables closer to its nominal value, but this can result steady state error (offset) in the output variables unless you have extra MVs at your disposal. In this case we would like to hold F_0 near its nominal value and let L_{1s} and L_{2s} freely vary.

The Rate Weight column penalizes changes on MV. The simultaneous objective is to minimize the weighted sum of controller adjustments, calculated as follows

$$S_{\Delta u}(k)=\sum_{i=1}^{M}\sum_{j=1}^{n_{mv}}\left\{w_j^{\Delta u}\Delta u_j(k+i-1)\right\}^2$$

Where n_{mv} is the number of manipulated variables, M is the number of intervals in the control horizon, $\Delta u_j(k+i-1)$ is the predicted adjustment in manipulated variable j and $w_j^{\Delta u}$ is the weight on this adjustment, is called the rate weight because it penalizes the incremental change rather than the cumulative value.
An increase of penalty on a given MV causes the controller to change it more slowly. As for the weight column, the entries here must be zero or positive real numbers. From figure 3.8 we hardly penalize changes in F_0, and allow sptL1 and sptL2 vary quite freely by setting their rate weight smaller. Contrary to weight, the rate weight values have no effect in steady state.

Output weight

The output weight tuning parameters let us dictate the accuracy within the desired output. It states the accuracy with which each output must track its set-point (or reference). This means that the controller predicts deviation over the prediction horizon. Each deviation is multiplied by the output's weight value, and the weighted sum of squared deviations $S_y(k)$ is computed as follows

$S_y(k)=\sum_{i=1}^{P}\sum_{j=1}^{n_y}\left\{w_j^y\left[r_j(k+i)-y_j(k+i)\right]\right\}^2$ Where n_y is the plant output, w_j^y is the weight for output j and the term $\left[r_j(k+i)-y_j(k+i)\right]$ is a predicted deviation for output j at interval $k+1$.
The weight must be zero or positive real number. One of the controller's objective is to minimize $S_y(k)$, thus a large weight on particular output causes the controller to minimize deviations in that output. In this case we want to minimize deviation in the output flow F_4, with that a large weight value on F_4 will be a natural choice.

Constraints

From figure A.5 clicking on the constraints tab we get the view shown in figure A.8. This pane allows us to set up constraints on manipulated variables and output variable (controlled variables). Constraints can be hard or soft; after creating the MPC controller from Simulink model all constraints are unconstrained by default.

Before specifying constraints, one should know that manipulated variables constraints are hard by default whereas controlled variable constraints are soft. This setting can be changed by using the constraints softening button in figure A.8.

The Constraints on manipulated variables in figure A.8 has 6 columns, where name and units columns are noneditable from here. To change them we have to use the signal definition view. The minimum and maximum value set each manipulated variable range. While max down rate and max up rate values states the amount each MV can change within a single control interval. The max down rate values must be negative or zero and the max up rate must be positive or zero. Furthermore the chosen constraints must be consistent with their nominal values (see figure A.5).
For constraints on output variables, the minimum and maximum value state each controlled variable range. Leaving the column empty mean to ignore constraints, the same can be done by editing – *inf* for a minimum or – *inf* for a maximum. The actually constraints values must be consistent we our nominal value (see figure A.5).

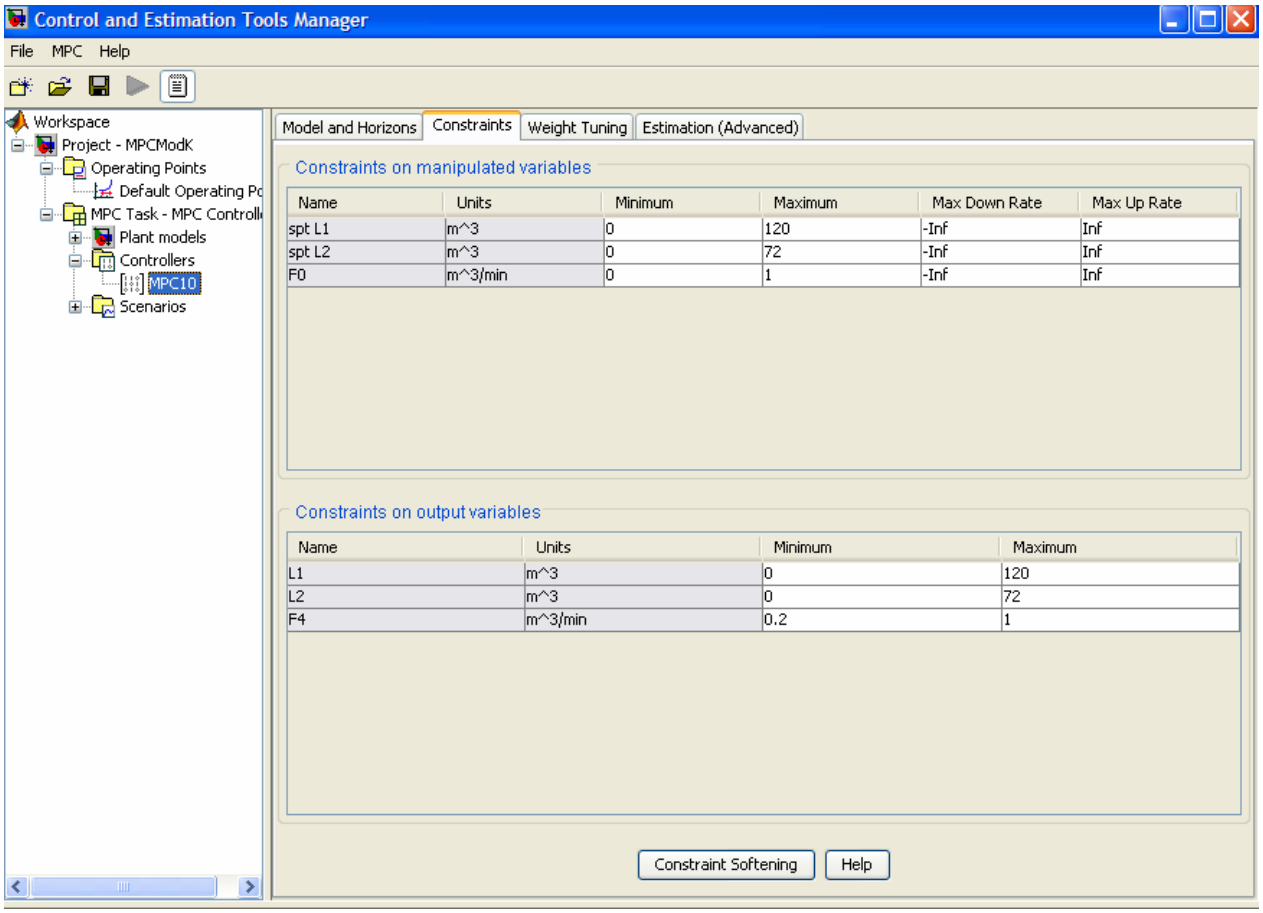

Figure A.8 constraints setting tab

A.3 How to run the Simulink model with MPC controller

Before you run the Simulink model with MPC control they are to things that have to be done. Considering that the Simulink model is open:

- You have to provide the designed MPC object to the workspace; this is achieved by loading the MPC object file in the Matlab command window.

Example: we want to use *MPC2* controller saved on *MPController2* (The MPC object name), in the command window enter:

* Load *MPContoller2*

- MPC2 controller is the in the workspace and read be used for the simulation

* Double click on the MPC controller block, then enter *MPC2* and run your simulation

To visualize how the MPC object is designed just inter controller name (e.g. *MPC2*)in the command window. The following MPC object view will appear on your computer screen.

```
MPC object (created on 12-Nov-2007 16:12:09):
---------------------------------------------
Sampling time:      1
Prediction Horizon: 120
Control Horizon:    2   3   5  10  15  20  30

Model:
        Plant: [3x3 ss]
      Nominal: [1x1 struct]
   Disturbance: []
        Noise: []

      Output disturbance model: default method (type "getoutdist(MPC2)" for details)

Details on Plant model:
                          --------------
    3  manipulated variables   -->|  6 states  |
                          |           |-->  3 measured outputs
    0  measured disturbances   -->|  3 inputs  |
                          |           |-->  0 unmeasured outputs
    0  unmeasured disturbances -->|  3 outputs |
                          --------------
Weights:
       ManipulatedVariables: [1.0000e-020 1.0000e-020 1.0000e-010]
   ManipulatedVariablesRate: [1.0000e-004 1.0000e-004 8000000]
          OutputVariables: [500 500 200000]
                      ECR: 8.0000e+011

Constraints:
 0 <= spt L1 <= 120, spt L1/rate is unconstrained, 0 <= L1 <= 120
  0 <= spt L2 <= 72, spt L2/rate is unconstrained,  0 <= L2 <= 72
      0 <= F0 <= 1,     F0/rate is unconstrained, 0.2 <= F4 <= 1
```

Appendix B: More cascade control

This appendix gives the Simulink model used for implementation of F3 and F2-F3 extra measurements using cascade control as well as the results obtained using this control scheme.

Cascade control structure (Models)

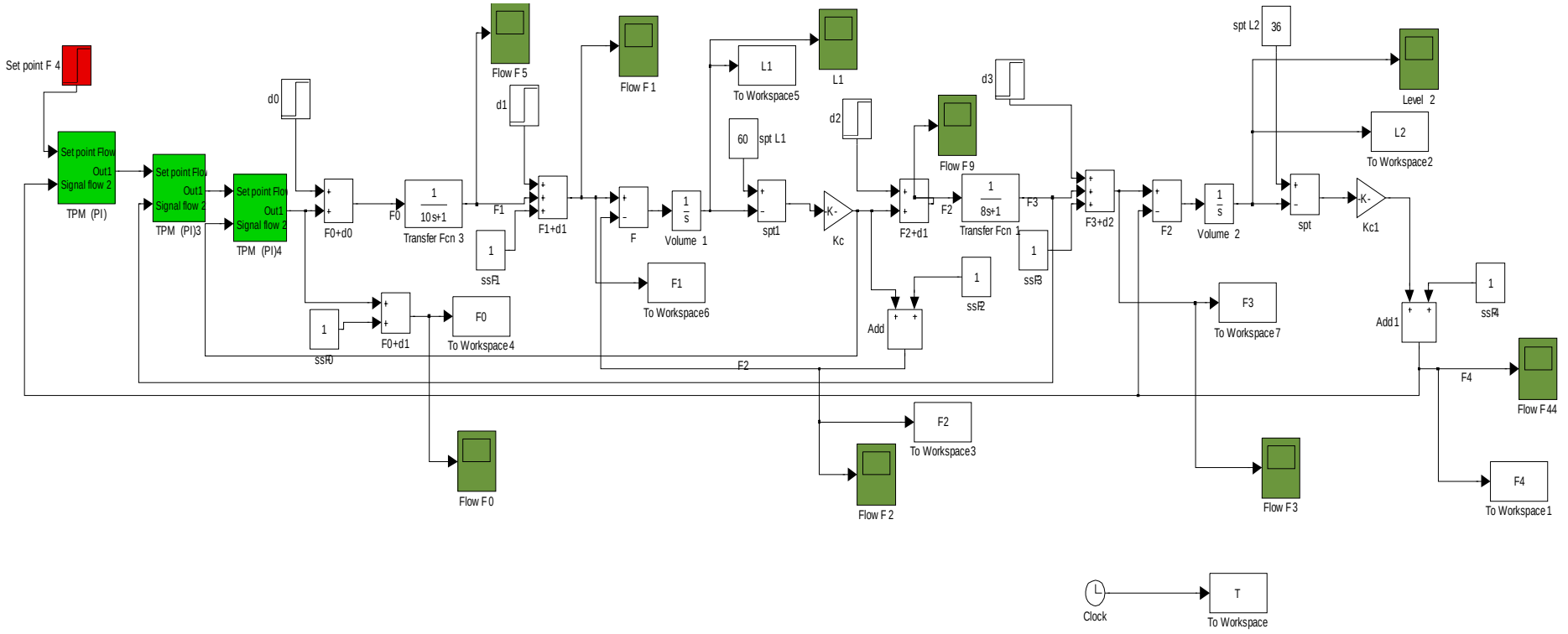

Figure B.0.1 Cascade control with F2-F3 as extra measurements

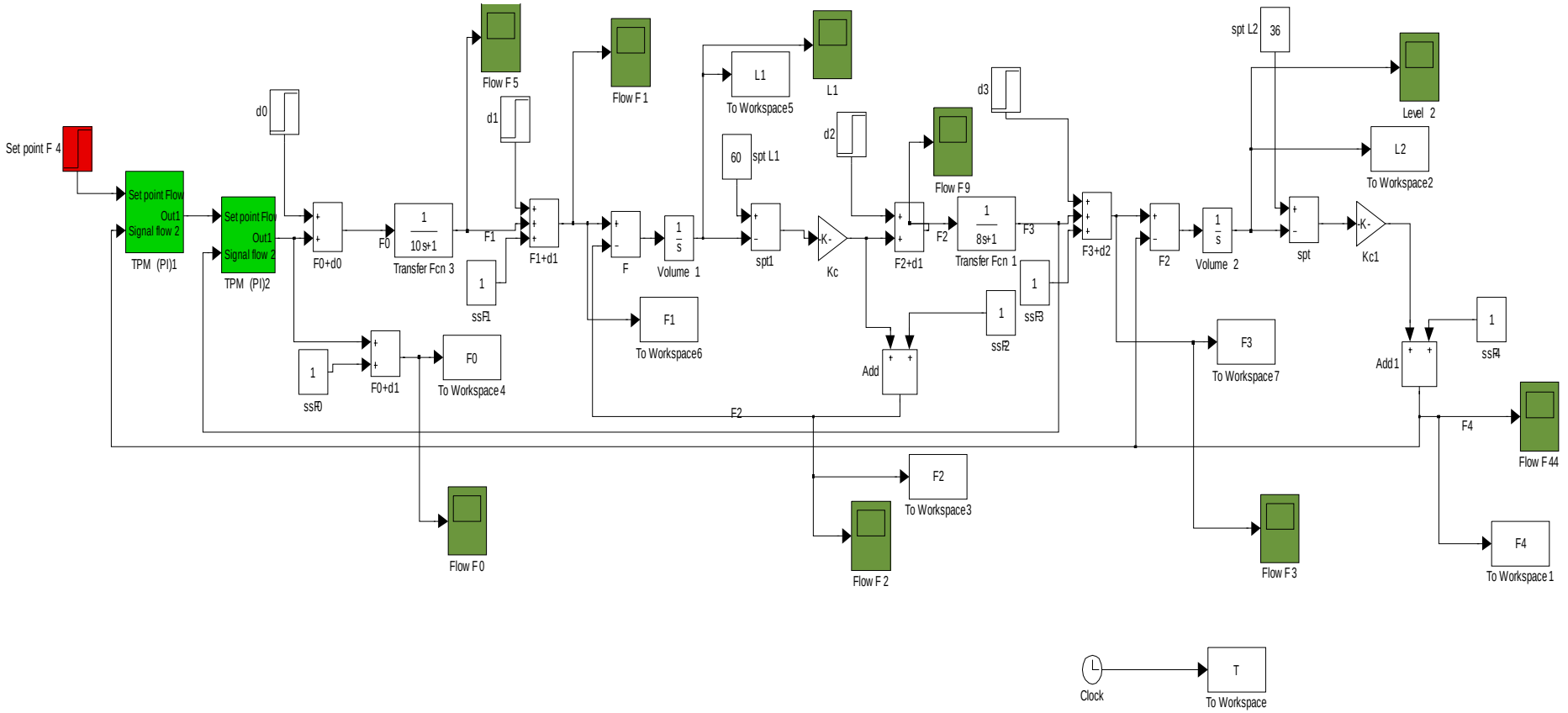

Figure B.0.2 Cascade control with F3 as extra measurements

Cascade control results

Extra measurement on F3
The models used to tune the PI controllers in figure B.2 were:

$$G_I = \frac{1}{25s+1}e^{-13s} \quad \text{(B.1)}$$

for the inner loop TPM (PI)2 and

$$G_O = \frac{1}{12s+1}e^{-26s}$$

for the outer loop TPM (PI)1 (B.2)

Extra measurement on F2 and F3
The models used to tune the PI controllers in figure B.1 ware:

$$G_I = \frac{1}{25s+1}e^{-5s} \quad \text{(B.3)}$$

for the inner loop TPM (PI)4 ,

$$G_O = \frac{1}{8s+1}e^{-10s}$$

for the next inner loop TPM (PI)3 (B.4)

$$G_O = \frac{1}{12s+1}e^{-20s} \quad \text{(B.5)}$$

for the outer loop TPM (PI)

Skogestad's tuning rules were used to tune the models above. See table B.1-B.5

Table B.1 tuning parameters for TPM (PI)4 controller in figure B.1

Tuning parameters	$\tau_c = \theta$	$\tau_c = 3\theta$
k_c	0.54	0.27
τ_I	26	26

Table B.2 tuning parameters for TPM (PI)3 controller in figure B.1

Tuning parameters	$\tau_c = \theta$	$\tau_c = 3\theta$
k_c	0.4	0.2
τ_I	8	8

Table B.3 tuning parameters for TPM (PI) controllers in figure B.1

Tuning parameters	$\tau_c = \theta$	$\tau_c = 3\theta$
k_c	0.3	0.15
τ_I	12	12

Table B.4 tuning parameters for TPM (PI)2 controller in figure B.2

Tuning parameters	$\tau_c = \theta$	$\tau_c = 3\theta$
k_c	0.96	0.66
τ_I	25	25

Table B.5 tuning parameters for TPM (PI)1 controller in figure B.2

Tuning parameters	$\tau_c = \theta$	$\tau_c = 3\theta$
k_c	0.23	0.15
τ_I	12	12

Results obtained using those implementations are shown as plots of controlled and manipulated variables in figure B.1 – B.12

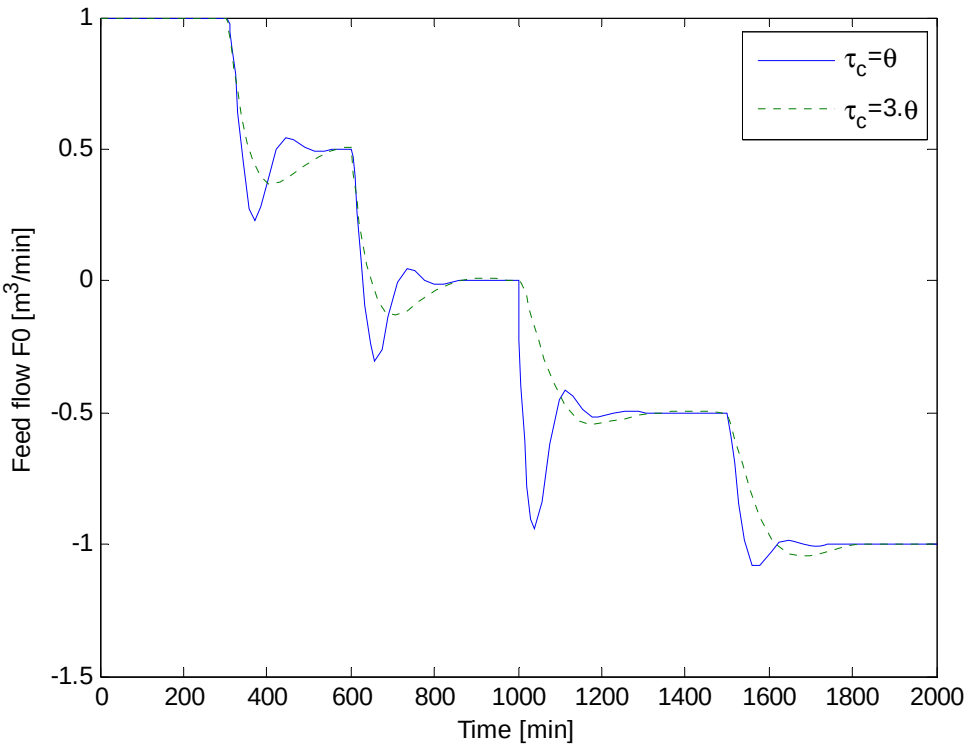

Figure B.0.3 Cascade control response F_0 using F_3 extra measurement with tuning $\tau_c = \theta$ (dashed) and $\tau_c = 3\theta$ (solid)

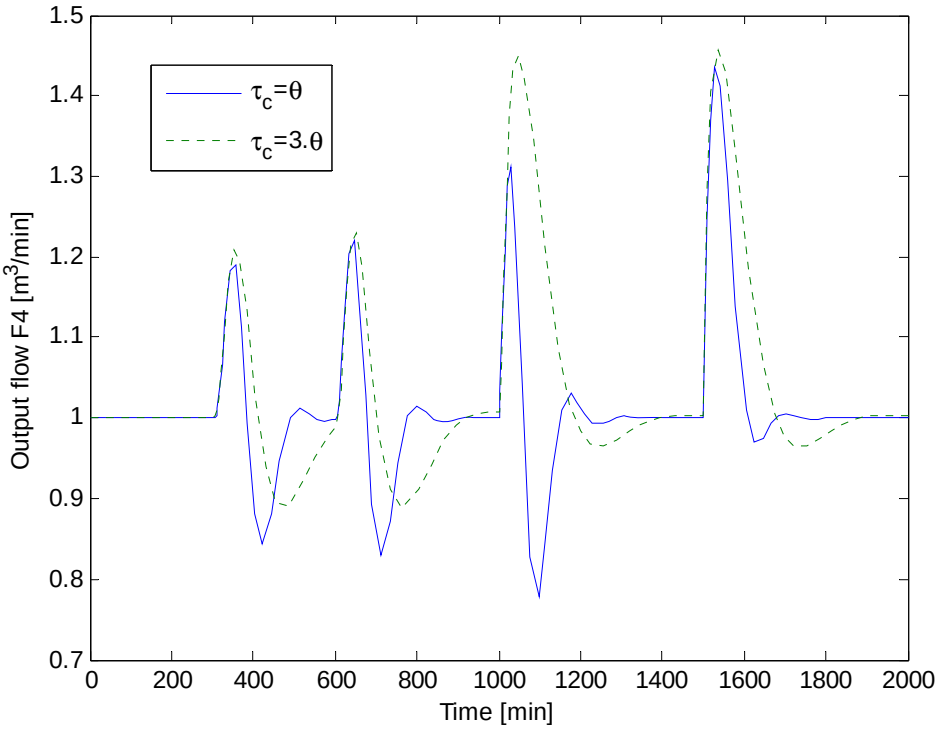

Figure B.0.4 Cascade control response F_4 using F_3 extra measurement with tuning $\tau_c = \theta$ (dashed) and $\tau_c = 3\theta$ (solid)

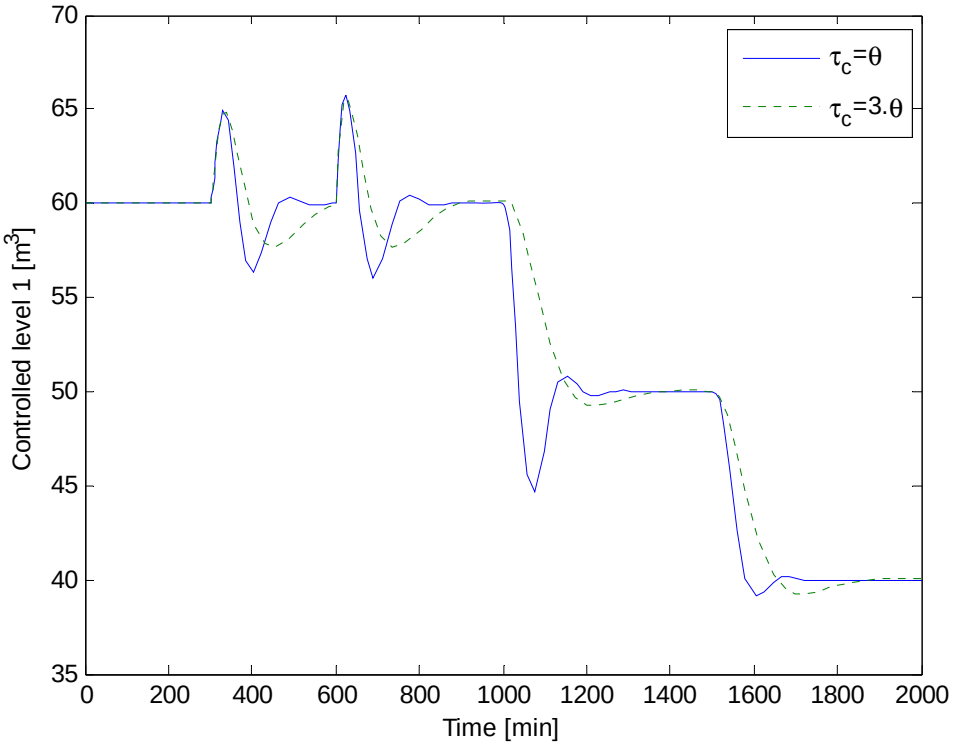

Figure B.0.5 Cascade control response L_1 using F_3 extra measurement with tuning $\tau_c = \theta$ (dashed) and $\tau_c = 3\theta$ (solid)

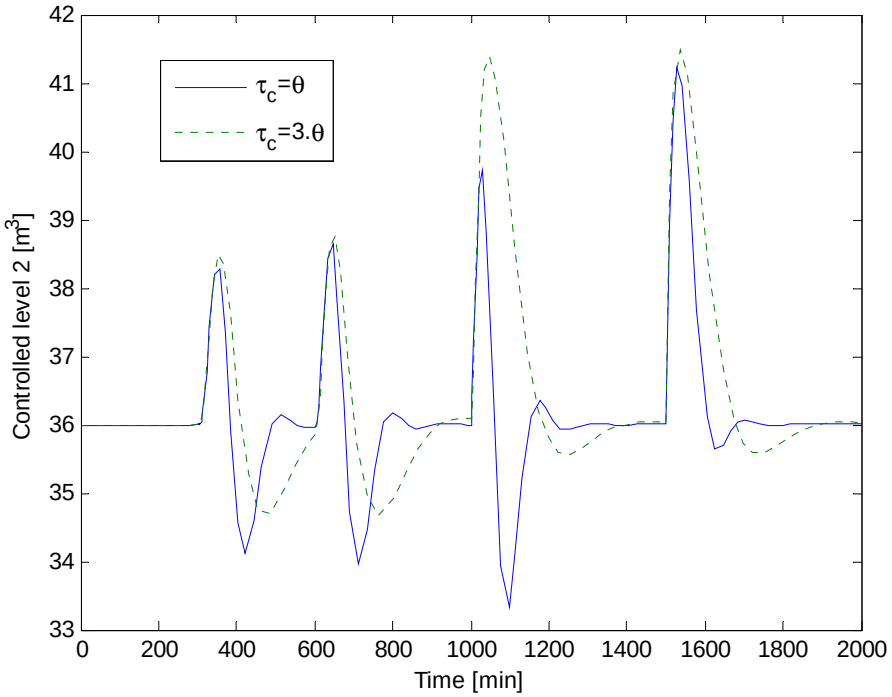

Figure B.0.6 Cascade control response L2 using F3 extra measurement with tuning $\tau_c = \theta$ (dashed) and $\tau_c = 3\theta$ (solid)

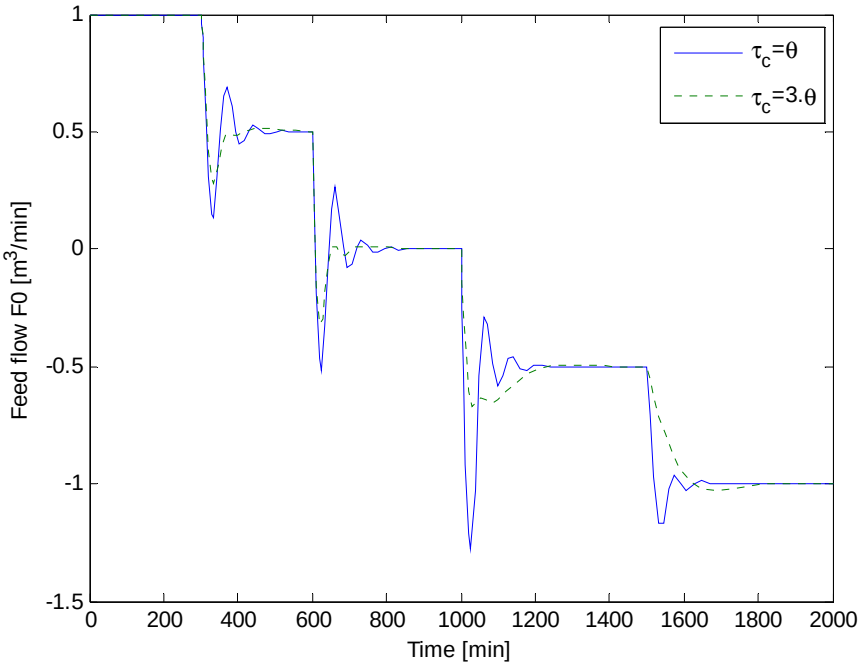

Figure B.0.7 Cascade control response F_0 using F_2-F_3 extra measurement with tuning $\tau_c = \theta$ (dashed) and $\tau_c = 3\theta$ (solid)

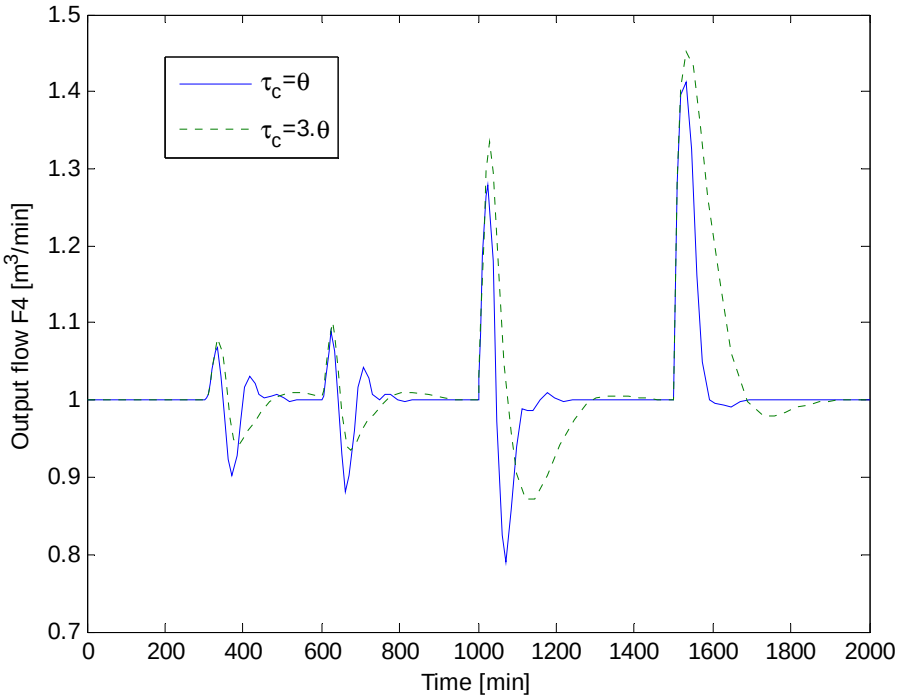

Figure B. 0.8 Cascade control response F_4 using F_2-F_3 extra measurement with tuning $\tau_c = \theta$ (dashed) and $\tau_c = 3\theta$ (solid)

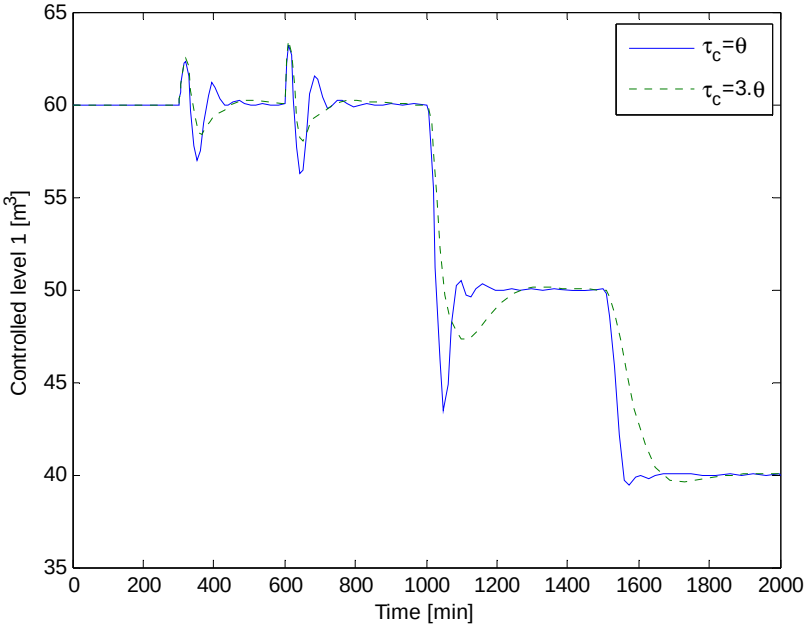

Figure B.0.9 Cascade control response L_1 using F_2-F_3 extra measurement with tuning $\tau_c = \theta$ (dashed) and $\tau_c = 3\theta$ (solid)

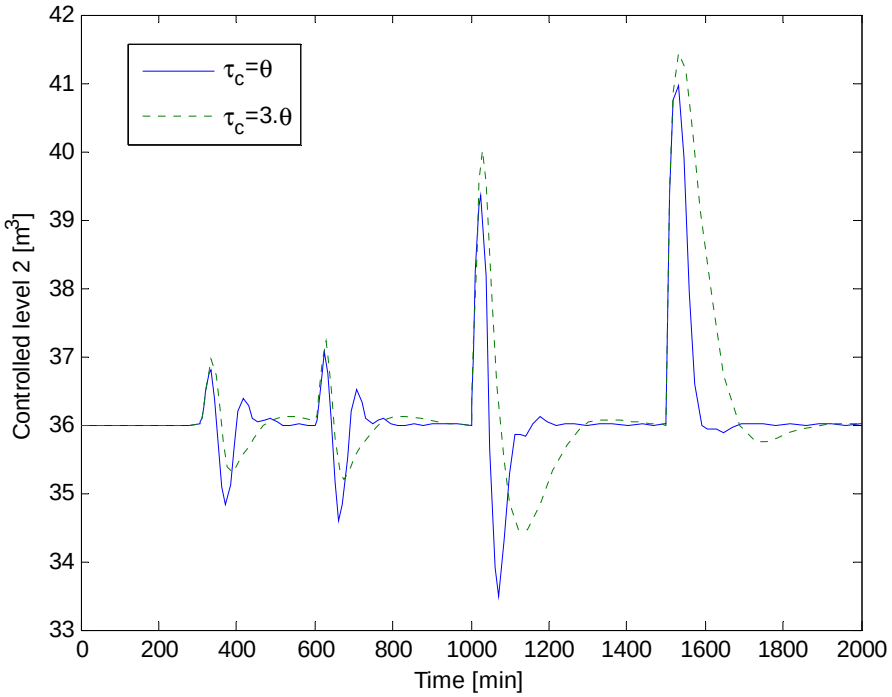

Figure B.0.10 Cascade control response L2 using F_2-F_3 extra measurement with tuning $\tau_c = \theta$ (dashed) and $\tau_c = 3\theta$ (solid)

Appendix C: Description of attached files

Single loop folder

ModelSL.mdl % Simulink model single loop control with $\tau_c = \theta$
ModelSL_c.mdl % single Simulink model loop control with $\tau_c = 3\theta$
Tauc=theta.mat % data for ModelsSL
Tauc=3theta.mat % data for ModelSL_c
Results.m % plots single loop control results

Single loop with feedforward folder

ModelSLFF_P.mdl % single loop with feedforward control with P controller on levels
ModelSLFF_PI.mdl % single loop with feedforward control with PI controller on levels
SingleloopFF_P.mat % data for ModelSLFF_P
SingleloopFF_PI.mat % ModelSLFF_PI
ResultsSLFF.m % Plots Results

Cascade control folder

Models with extension 'c' were tuned using $\tau_c = 3\theta$ while those without this extension were tuned using $\tau_c = \theta$

ModelcascadeF2F4.mdl
ModelcascadeF3F4.mdl
ModelcascadeF2F3F4.mdl
ModelcascadeF2F4c.mdl
ModelcascadeF3F4c.mdl
ModelcascadeF2F3F4c.mdl
The following files contain data for the Simulink models above
casF2_F4tauc=theta.mat
casF2_F4tauc=3theta.mat
casF3_F4tauc=theta.mat
casF3_F4tauc=3theta.mat
casF2_F3_F4tauc=theta.mat
casF2_F3_F4tauc=3theta.mat
ResultsCascade.m % plots results

MPC folder

MPCModel.mdl % Simulink model for MPC
MPCobject1.mat % MPC object MPC1 controller
MPCobject2.mat % MPC object MPC2 controller
MPC1data.mat % Data from MPC1
MPC2data.mat % Data from MPC2
MPCResults.m % Plots results from MPC

Comparison of results folder

Tauc=theta.mat % Single loop control
SingleloopFF_P.mat % Single loop with feedforward control
casF2_F4tauc=theta.mat % Cascade control
MPC1data.mat % Model Predictive Control
CompResults % Plots results