NTNU
Norges teknisk-naturvitenskapelige universitet
Fakultet for naturvitenskap og teknologi
Institutt for kjemisk prosessteknologi

SPECIALIZATION PROJECT 2013

TKP 4550

PROJECT TITLE:
Modelling the bulk cooler in NPK 4 as a series of CSTRs

By
Bjørn Tore Mathisen

Supervisor for the project: Sigurd Skogestad, Jakub Bujalski

Date:10/12/2013

Abstract

The following report is a 5th year individual project submission. As part of the masters program at the process systems engineering group at NTNU. The project had a rough start (see section D), dealing with sever software issues already at the early stages. The added support and understanding of the supervisors Jakub Bujalski and Sigurd Skogestad was therefore greatly appreciated through the many frustrating moments. Also a big thank you to Volker Siepmann for several helpful and interesting modelling discussions and for his dedication in trying to resolve mentioned software issues. The projects intention is to build a platform for modelling the solid flow in NPK factory 4 at Yara Porsgrunn. This model comprehends the bulk cooler in NPK 4 which was installed the summer of 2013.

The bulk cooler in NPK 4 is a vertical plate heat exchanger transporting heat between a solid and a liquid. The liquid water flows inside the plates and the solid NPK fertilizer is transported between the plates on the shell side. To keep the fertilizer in the heat exchange zone a constant level of NPK is kept by controlling the outlet valve situated in the bottom. The bulk cooler has two sections, a co-current unit and a counter current unit where the counter current is situated on top of the co-current. The liquid enters the bulk cooler at the co-current stage. See figure 10 which was constructed in reference to figure 34 in section B.

A small amount of data regarding the physical properties for different NPK specification resulted in an attempt to approximate a generic Cp_{NPK} suitable for the model. However, as seen in section B.1, the existing data only comprehend high N - NPK products and also reveals the appearance of a phase change as the ammonium nitrate restructures it's crystals. This increased complexity was handled by modelling the phase change over the span of $1°C$

The bulk cooler was modeled as seen in figure 10. This modelling scheme resulted in a system of CSTR's where the CSTR's placed in series approximates the temperature distribution throughout the bulk cooler. To evaluate each sections energy balance, MATLAB was used in coherent with Simulink, where the differential equations were handled by a Simulink s-function. The resulting model did not fully correspond to the design specifications, see figure 12. Therefore two parameters was introduced; $p.alfa$ adjusts the effective heat exchange area as the solid are packed spheres with a following void fraction, and $p.beta$ adjust the effective specific heat capacity. The adjustment of the heat capacity is justified by looking at section B.2, which reveals that the highest N containing species in general have the highest heat

capacities. Which leads to adjusting with $p.beta$ for high potassium containing products like NPK $16-16-16$.

The fitted model was then tested dynamically after introducing a PI controller. When increasing the inlet flow and temperature towards the design rates for the $250 ton/h$ scenario (figure 31) , the model was able to control the temperature of the NPK (figure 26). At the same time it failed to sufficiently describe the liquid phase. This indicates that the parameters $p.alfa$ and $p.beta$ is invalid for flow rates sufficiently larger than the design rates.

A comparison between the model and plant data was initiated but later terminated due to too unreliable process data. See section 4.6.

Suggested further development include (but is not limited to); laboratory experiment to determine specific heat capacity for the range of NPK products; code optimization and help section addition, and also a plant test run, to fully determine the outlet-inlet temperature correlation in reference to the flow rates.

Contents

1 Introduction

2 General process Description

The Yara NPK 4 factory is situated at Herøya Industrial Park in Porsgrunn. Here it produces NPK fertilizer by using the products of near by factories. NPK fertilizer is a product that mainly contains Nitrogen, Phosphor and Potassium. The three core nutrition's for the plant life. The main raw products of the process are nitric acid, ammonia, ammonium nitrate, potassium salts and calcium phosphate. Where everything but the phosphate and the potassium are manufactured by neighbouring Yara plants on site.

2.1 Pre NPK 4

The first section of the product line starts in NPK 4's neighbour factories, NPK factory 2 and 3. Here the raw phosphate stone (a calcium phosphate containing compound) enters a reactor. The raw phosphate is highly insoluble and nitric acid with the highest purity (60%) is required to solve the phosphate. The calcium reacts to form $CaNO_3$ which is used in calcium nitrate based fertilizers (which is also produced by Yara in Porsgrunn). The calcium nitrate is crystallized by cooling with sub zero ammonia-water liquid and separated by use of rotary vacuum-drum filters. The remaining mother lite is stored for use in NPK 4.

2.2 Neutralization

The mother lite contains a sever amount of phosphor and nitrogen but is yet to be in a state that is applicable for agriculture. The flow entering NPK 4 has a very low pH value as it contains mostly H_3PO_4 and HNO_3. While being rich on N and P the mother lite needs to be neutralized to a state suitable for growing crops. The acid enters NPK 4 and is corrected with high N containing substances as ammonium nitrate and nitric acid to reach the desired N/P relationship. A relationship which is depending on the product specification. The corrected stream, which still contains remnants of calcium nitrate then enters the main reactor. The reactor is a circulating pressurized (2.1 bar) pipe reactor where the acid enters in the bottom as a liquid. It then proceeds upwards where it reaches an ammonia distributor where the entering ammonia in gas phase reacts with the acid and neutralizes the solution. The net exothermic reactions (particularly exothermic in the formation of AN) boils a off a lot of the water in the mother lite. This in turns yields a two phase stream with a net low density which raises to the top of the reactor and reaches a separator. Here the gas phase is separated by use of the difference in density in relation to the centripetal force. The high density liquid circulates back to the bottom of the reactor towards the inlets.

This part of the flow also describe the driving force of the circulation of the reactor, the high density liquid pushes the low density two phase stream upwards on the other side. The resulting main neutralization reactions are as follows:

$$HNO_3 + NH_3 \rightarrow NH_4NO_3 \tag{1}$$
$$H_3PO_4 + NH_3 \rightarrow NH_4H_2PO_4(MAP) \tag{2}$$
$$NH_4H_2PO_4 + NH_3 \rightarrow NH_4)_2HPO_4(DAP) \tag{3}$$

For some of the additional reactions the parameter Ca/P has a big impact. Small amounts of calcium nitrate which comes with the mother lite, forms a water soluble calcium phosphate in the reactor. A water soluble component is easily acquirable for the plants. These calcium phosphate components are also valuable for the plant as calcium in it self is also an important plant nutrient. In addition to the water soluble $Ca-P$ components, some acid soluble substances are also formed, these will be available for the plants over time. Preserving some of the fertilizer to give an effect over a longer timescale. However, too much calcium will result in in acid insoluble components and for the farmer that's the equality of spreading concrete over his field. For Yara it's a waste of valuable phosphor as additional phosphor components must be added to compensate. In general, phosphor is the most expensive nutrient in NPK

The reactor is pressurized so it is easy to exchange the energy in to 1 *baro* steam. Partly the steam is used to heat up a significant amount of households and offices in the Porsgrunn area.

2.3 Evaporation

After the reactor the water content of the solution containing N and P is at about 15% and needs to be lowered to 0.5%. The water content must be constrained because of storage and fertilizer quality implications. The water is evaporated by use of two heat exchangers operating at a low pressures, close to vacuum, on the tube side. On the shell side there is high pressure (18 *baro*) steam condensing on the exterior of the internal pipes. After each exchanger a separator removes the gas phase. In the first evaporator the solution enters from the bottom and travels vertically through the entirety of the exchanger. The second evaporator is different from the first in the way it functions. In this heat exchanger the liquid enters on the top of the heat exchanger, and travels like a film om the outside of the pipes downwards. This results in an heat exchanger with instant gas/liquid separation and a net shorter hold up time, which prevents some particles from solidifying.

2.4 Mixing and Prilling

The last macro nutrient, P is added by mechanically mixing the hot NP solution with fine potassium containing salts. In addition some solid NPK particles are added for two purposes; one - recirculate product outside of the quality constraints, two - control the temperature of the slurry. The two types of potassium salts used in the process are KCl and K_2SO_4 which one used has a direct impact on the mixers set point for temperature. Controlling the temperature is of the utmost importance, since it has to be directly above where the slurry solidifies. The result of the mixing is a slurry which is directed to a centrifuge operating with approximately 450 rotations per minute. The centrifuge is situated on the very top of a 60 m open tower and slings the liquid through small holes forming hundreds of droplets raining down and solidifying along the way as it's cooled by the air.

2.5 Dry section

After the prilling stage three main objectives are yet to be completed. The first task is to make sure the fertilizer size distribution is within the required specifications. This is important as the prills must fit the standardised distribution machines used by the local farmers. Generally the fertilizer should have a diameter in the area between 2-4mm. This is achieved by having 4 sieving trays in parallel to first remove everything + 4mm and then another 4 sieving trays to remove the sub 2mm. $+4mm$ is crushed to smaller particles and re enters the sieving trays. The sub $2mm$ is resend to the mixer and re prilled.

The NPK then needs to be cooled below $32°C$ to make sure that the NPK don't harden in storage and clump together. This is because ammonium nitrate goes through a phase change in the vicinity of $33°C$, restructuring it's crystals. This cooling of the NPK was until the summer of 2013 done solemnly by 2 fluidized beds in parallel. The units fluidized and cooled the NPK by outside air sucked through the beds. During the summer, the fluidized beds became a major production bottleneck as the outside air was to hot for sufficient cooling. Cooling the air directly is also limited as the problem of water condense arises. The result was the creation of the bulk cooler which this project attempts to model.

2.5.1 Bulk cooler description

The bulk cooler is a vertical plate heat exchanger between liquid water and solid NPK particles in two sections. To get a better impression, below (figure 1) is a sketch made by the manufacturing company Solex thermal science [5]. The heat exchanger in NPK 4 has two heat exchange sections situated directly above each other but is otherwise identical.

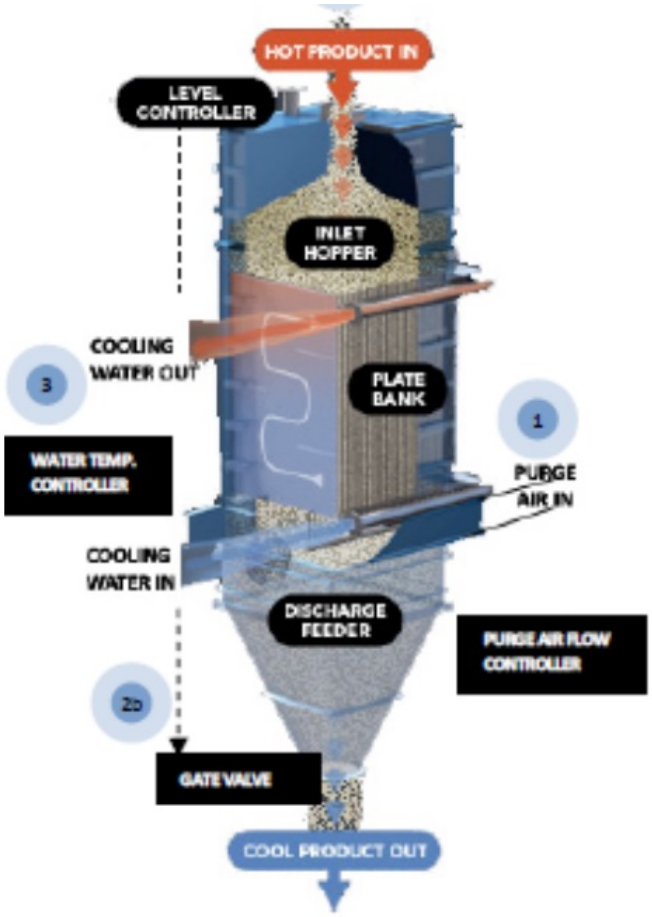


Figure 1: Sketch of a Solex bulk cooler unit[5]

The water flow is distributed inside the plates while the NPK fills the shell side by use of a level controller keeping the NPK level above the heat exchange area. Figure 1 shows a single counter current solution. The NPK 4 bulk cooler has a mixed set up. The cooling water enters the bottom heat exchanger and flows co-current to the bottom. The water then flows back up to the top heat exchanger and flows counter current trough that one. For a clearer view a flow sheet is included in section B.6.

The NPK is transported as bulk. To protect the prills from humidity aminol and talcum powder enters a drum with the prills. Giving them a thin protective coating before entering the storage facilities, awaiting transport by ship.

3 Modelling in MATLAB

3.1 Fitting a expression for calculating Cp NPK

Yara supplied an enthalpy diagram for NPK grade 21-4-10 as can be seen under section B.1. The data needs to be fitted to a continuous function usable by MATLAB. The standard representation of the temperature correlation with the specif heat capacity is a polynomial on the form of:

$$Cp_i^s = a_{i1} + a_{i2}T + a_{i3}T^2 + a_{i4}T^3 \tag{4}$$

Integrating the equation yields an expression for the enthalpy:

$$h_i^s = \Delta_f h_i^s + \int_{T_ref}^{T} Cp_i^s dT \tag{5}$$

$$h_i^s = K_1 + a_{i1}T + \frac{a_{i2}}{2}T^2 + \frac{a_{i3}}{3}T^3 + \frac{a_{i4}}{4}T^4 \tag{6}$$

The idea is to create a sub function that at all time can calculate the current cp for each time step.

3.1.1 Linear approximation

For the early design stage a linear approximation is assumed sufficient. By studying the enthalpy diagram B.1 it's easy to spot that there is a clear miss match between a linear approximation and the enthalpy change in the temperature range of the NPK cooler ($43°C \rightarrow 32°C$). A vertical drop implies a phase change and by checking the species enthalpy diagram in section B.2 it is clear that the Ammonium Nitrate (NH_4NO_3) goes through a phase change, restructuring the AN crystals . But for the early models a linear approximation is assumed sufficient, since the main goal for the early stage is hitting the output temperatures. Describing the interior temperatures in detail is not as important at this point. Therefore $a_{i2} \rightarrow a_{in}$ is set to zero, leaving the equation at:

$$h_i^s = K_1 + a_{i1}T \tag{7}$$

Extracting data from the enthalpy diagram:

$$a_{NPK1} = \frac{32 - 9,5}{42 - 32} kJ/kgC^\circ = 2250 kJ/kgC^\circ \tag{8}$$

3.1.2 Fitting higher order polynomials

For approximating the enthalpy change by higher order polynomials the following method is chosen; several data points will be graphically extracted from the enthalpy diagram in B.1 and then multiple polynomials of varying order will be established and the best approximation chosen. The extracted data points are presented in the following table:

Point	Enthalpy $[KJ/kg]$	Temperature $[c^{\circ}]$
1	50	54
2	32	42
3	29	40
4	21	34
5	13	34
6	9.5	32
7	0	25

Table 1: Enthalpy data points for NPK 21-4-10

To try to fit an expression with this data the MATLAB function polyfit was used, see appendix **??**:

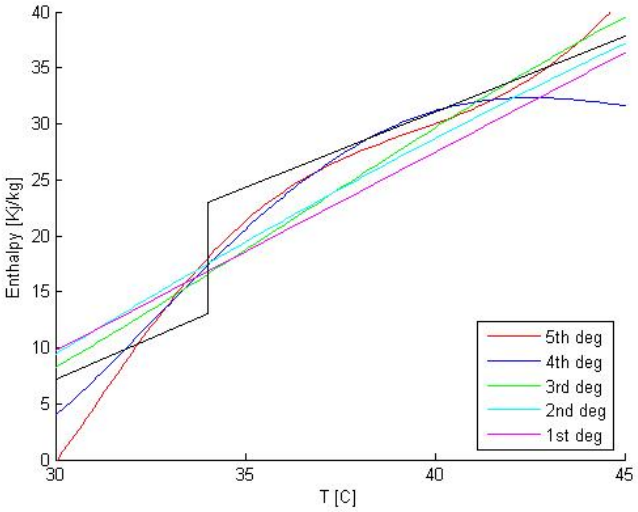


Figure 2: Temperature Enthalpy diagram for NPK 21-4-10 and fitted polynomials from 1st -5th degree

As can be seen on the figure, the polynomial approximation is badly conditioned over the phase change and will not significantly improve the model in respect to additional computing. Using a 5th degree polynomial achieved what was considered as the best result for the data selection in the design temperature range ($43°C \rightarrow 32°C$).It was attempted to experiment with higher order polynomials and/or adding additional data points. But high order polynomials resulted in very badly conditioned functions and chasing higher order polynomials leads to an increase in the required data points as ($n^{max}_{p.degree} = n_{datapoints} - 1$) [6]. Which in turn is problematic since it is necessary to consider the clear trade off from adding additional data points and the accuracy which these can be obtain by graphical reading of values from the diagram in B.1.

3.1.3 Phase change over the span of a $\Delta T = 1$

Another way of creating an approximation of the phase change is by modelling the phase transition over the range of a one degree temperature span. Basically this models the phase change as three-split linear function. By using the data in 3.1.2 the following model was created:

$$a_{NPK1} = \frac{13 - 0}{33 - 25} * 1000 J/kgC^{\circ} \qquad for\ T < 33C^{\circ} \tag{9}$$

$$a_{NPK1} = \frac{23 - 13}{34 - 33} * 1000 J/kgC^{\circ} \qquad for\ 34 < T < 33C^{\circ} \tag{10}$$

$$a_{NPK1} = \frac{50 - 23}{54 - 34} * 1000 J/kgC^{\circ} \qquad for\ 34 < TC^{\circ} \tag{11}$$

Which yields:

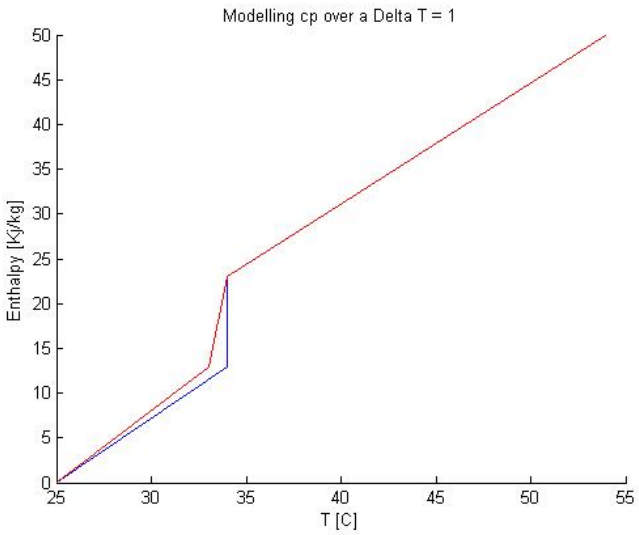


Figure 3: First attempt on modelling the phase change by use of a three split linear function

Adjusting the values for a better fit:

$$a_{NPK1} = \frac{12-0}{33.5-25} * 1000J/kgC^{\circ} \qquad for\ T < 33.5C^{\circ} \qquad (12)$$

$$a_{NPK1} = \frac{24-12}{34.5-33.5} * 1000J/kgC^{\circ} \qquad for\ 34.5 < T < 33.5C^{\circ} \qquad (13)$$

$$a_{NPK1} = \frac{50-24}{54-34.5} * 1000J/kgC^{\circ} \qquad for\ 34.5 < TC^{\circ} \qquad (14)$$

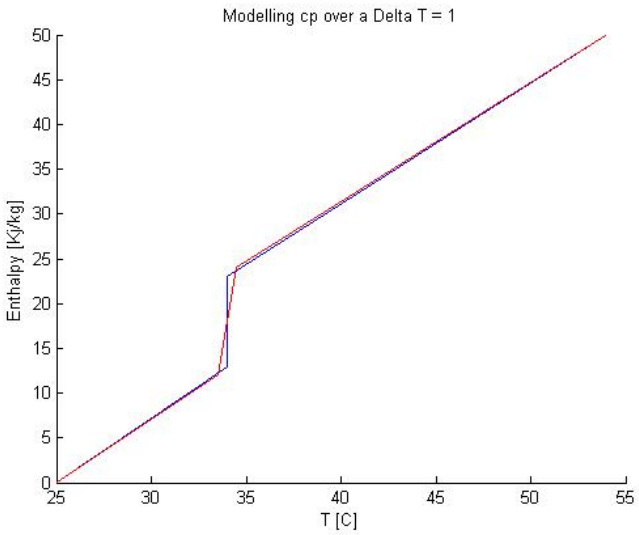


Figure 4: Adjusted model of the phase change by use of a three split linear function

3.2 Model with simplified lumped systems (CSTR approach)

The idea is to model the entire bulk cooler as a single CSTR. Completely ignoring the the interior temperature gradient, but trying to hit the design outlet temperature when given the design inlet temperature. This first simple model has the following topology:

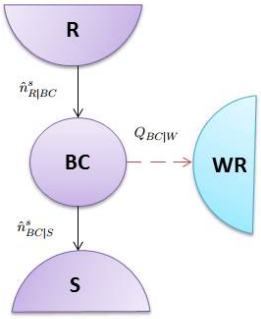


Figure 5: Topology of the bulk cooler model with a simple CSTR approach

Where R denotes reservoir, the topology term for the NPK supply and the S denotes NPK storage facility, which is also modeled as a reservoir (assuming infinite capacity). WR is noted as the cooling water reservoir. Notice that this approach models the heat exchange directly to the WR reservoir as the cooling water is modelled to have a constant temperature and therefore a total unlimited heat capacity. This simplifies the model as there are no balances for the liquid side of the bulk cooler.

3.2.1 Mass balance

[1]

$$\frac{dN^s_{BC}}{dt} = \hat{n}^s_{R|BC} - \hat{n}^s_{BC|S} \tag{15}$$

3.2.2 Energy balance

By neglecting the kinetic and potential energy the following energy balances is established:

$$\frac{dU^s_{BC}}{dt} = \hat{H}^s_{R|BC} - \hat{H}^s_{BC|S} - Q_{BC|W} \tag{16}$$

Introducing a general simplified heat exchanger model:

$$Q_{BC|W} = UA\Delta T \tag{17}$$

Where ΔT is set to the difference between the average temperature of the design data for the cooling water and the temperature of the bulk cooler:

$$\Delta T = \frac{T_{c,in} - T_{c,out}}{2} - T_{BC} \tag{18}$$

Introducing the enthalpy:

$$H = U + pV \tag{19}$$

$$\frac{dH}{dt} = \frac{dU}{dt} + V\frac{dp}{dt} + p\frac{dV}{dt} \tag{20}$$

Resulting in the following enthalpy balance:

$$\frac{dH^s_{BC}}{dt} = \hat{H}^s_{R|BC} - \hat{H}^s_{BC|S} - Q_{BC|W} \tag{21}$$

The system is under a constant atmospheric pressure, resulting in the enthalpy being a function of T, n and has the following total differential:

$$\frac{dH^s_{BC}}{dt} = \frac{\partial H^s_{BC}}{\partial T_{BC}}\frac{dT_{BC}}{dt} + \frac{\partial H^s_{BC}}{\partial n_{BC}}\frac{dn_{BC}}{dt} \tag{22}$$

$$\frac{dH^s_{BC}}{dt} = n_{BC}c_p(T_{BC})\frac{dT_{BC}}{dt} + h(T_{BC})\frac{dn_{BC}}{dt} \tag{23}$$

Where the heat capacity and the specific enthalpy is a function of the temperature. Inserting 23 in the energy balance and introducing the mass balance 15.

$$\begin{aligned} N_{BC}c_p(T_{BC})\frac{dT_{BC}}{dt} &= (h(T_R) - h(T_{BC}))\hat{n}^s_{R|BC} - (h(T_{BC}) \\ &\quad - h(T_{BC}))\hat{n}^s_{BC|S} - Q_{BC|W} \end{aligned} \tag{24}$$

Computing the differences in the partial molar enthalpies by:

$$h(T_a) - h(T_b) = \int_{T_b}^{T_a} c_p(T)dT \tag{25}$$

Integrating the the c_p relation with the linear approach from section 3.1.1 yields:

$$h(T_a) - h(T_b) = [K_1 + a_{i1}T]_{T_b}^{T_a} = (T_a - T_b)a_{i1} \tag{26}$$

$$\frac{dT_{BC}}{dt} = \frac{1}{N_{BC}a_{NPK1}}\left((T_R - T_{BC})a_{NPK1}\hat{n}^s_{R|BC} - Q_{BC|W}\right) \tag{27}$$

3.2.3 Using steady state design data to evaluate the bulk cooler heat transfer coefficient

To evaluate the heat transfer coefficient in the bulk cooler the steady state values can be used to explicitly solve the energy balance in respect to the heat transfer coefficient.

$$0 = \left| \frac{1}{N_{BC} a_{NPK1}} \Big((T_R - T_{BC}) a_{NPK1} \hat{n}^s_{R|BC} - UA\Delta T \Big) \right|_{ss} \tag{28}$$

$$U = \frac{1}{A\Delta T} (T_R - T_{BC}) a_{NPK1} \hat{n}^s_{R|BC} \tag{29}$$

3.2.4 State identification

The two states is at this point quite obvious and will be set to be the mass and temperature of the NPK in the bulk cooler. The input (or rather the disturbance) of the system is the mass flow rates of the NPK. When it comes to process constant for this early and simplified model the cooling water temperature will be set as one of them. This should be a fair early simplification since the flow rate and the temperature of the cooling water is controlled. This leads to the following state space representation:

$$\underline{x} = \begin{bmatrix} N^s_{BC} & T_{BC} \end{bmatrix}^T \tag{30}$$

$$\underline{u} = \begin{bmatrix} \hat{n}^s_{R|BC} & \hat{n}^s_{BC|S} & T_R \end{bmatrix}^T \tag{31}$$

$$\underline{\gamma} = \begin{bmatrix} (T_{c|in} - T_{c|out})/2 \end{bmatrix}^T \tag{32}$$

$$\underline{\Theta} = \begin{bmatrix} a_{NPK,1} & UA \end{bmatrix}^T \tag{33}$$

Resulting in:

$$\frac{d\underline{x}}{dt} = \begin{bmatrix} u_1 - u_2 \\ \frac{1}{x_1}(u_3 - x_2)u_1 - \frac{\Theta_2}{x_1 \Theta_1}(x_2 - \gamma_1) \end{bmatrix} \tag{34}$$

3.2.5 Simulink model of the dynamic system

The simple CSTR type model will by default be correct for the steady state system, since it has been solved by choosing the appropriate value for U. Therefore, the model validation must the be achieved through testing with disturbances that shifts the balances away from steady state. For such a purpose Simulink is an excellent tool. If set up correctly Simulink will provide an easy to follow graphical representation of the system where tweaks and testing are easy to perform. To initiate the simulation and present the data a simple MATLAB script was established, see section A.1. Then a Simulink model was created (see figure 6) and the differential equations was solved by use of the S-function block:

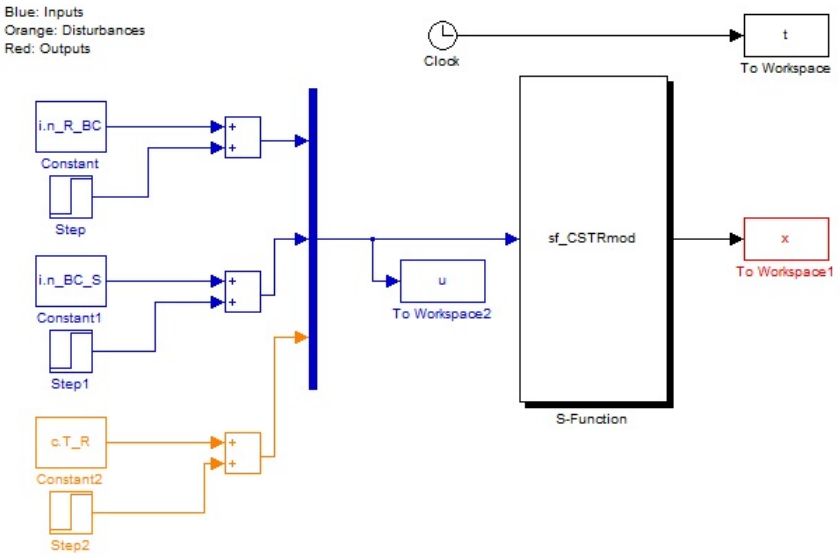


Figure 6: Simulink model of the simple "CSTR" model

See section A.1.3 for the interior code of the S-function block.

A series of step tests was conducted for the single CSTR model. One for each disturbance/input. A 10% step in the inlet temperature yields:

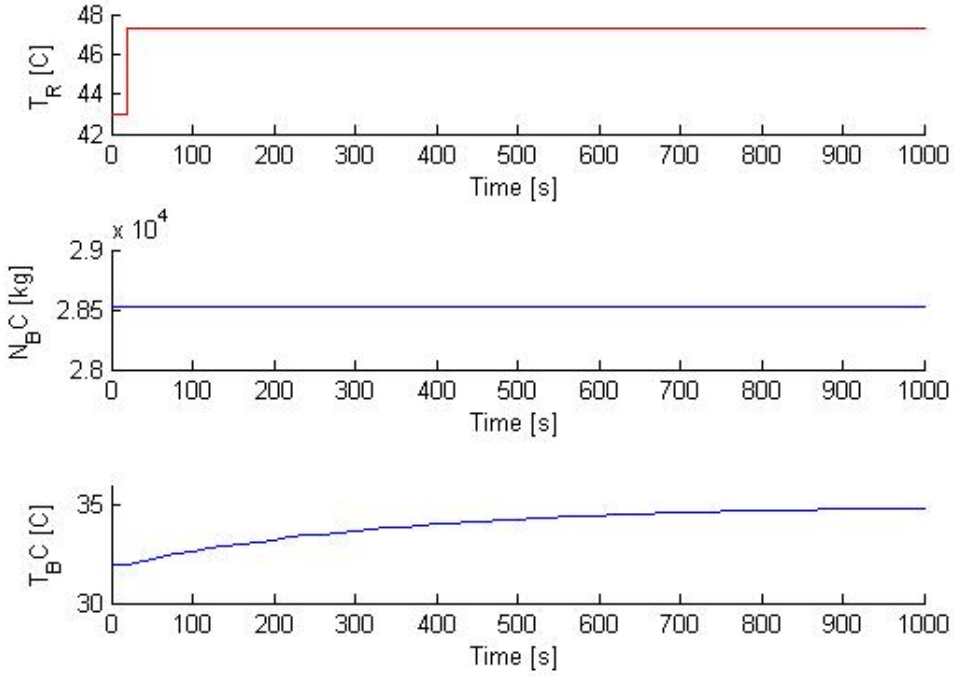


Figure 7: Single CSTR model's step response for 10% increase in inlet temperature for the NPK

10% step in the inlet flow of NPK:

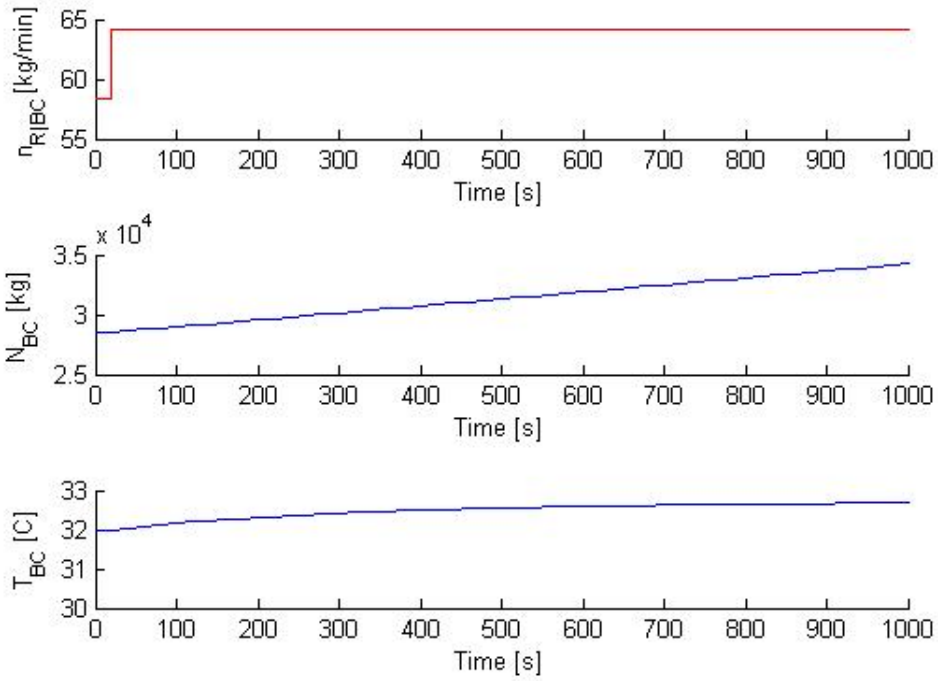


Figure 8: Single CSTR model's step response for 10% increase in inlet flow of NPK

10% step in the outlet flow of NPK:

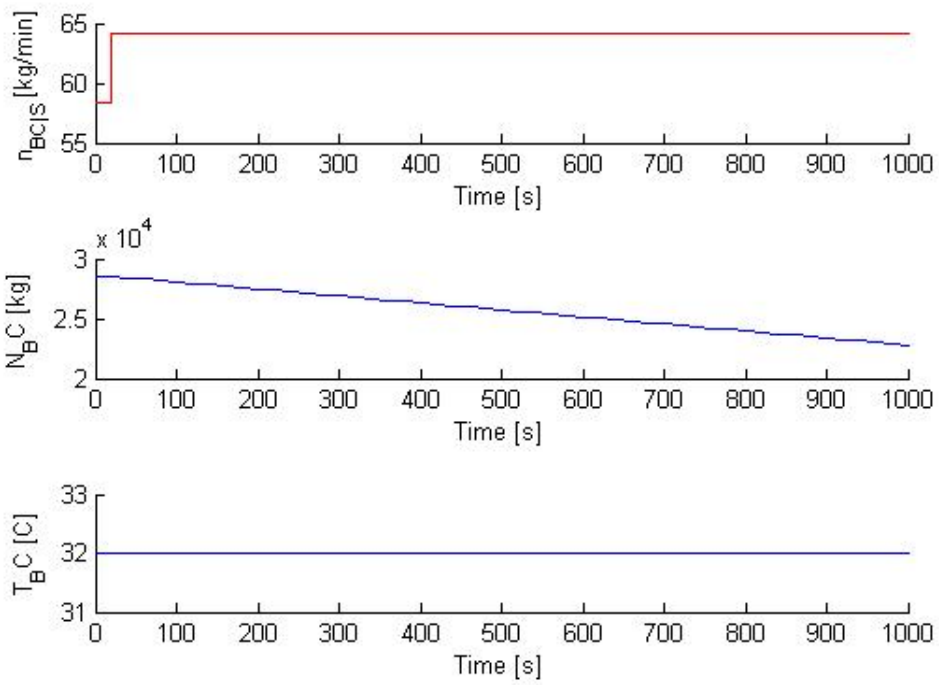


Figure 9: Single CSTR model's step response for 10% increase in outlet flow of NPK

At this point it becomes transparent that the first order responses over simplifies the dynamic behaviour of the bulk cooler. Since it is expected that the mixed co-current and counter current heat exchanger has a higher complexity in regards of it's transfer function. However, the method seems promising and some of the modelling work will be used further in section 3.3

3.3 Model with lumped systems in series

The model divides the bulk cooler in to several section where an arbitrary interior section can be describe by using balances similar to the CSTR in 3.2. The set of sections is then divided in to two sets. One top section handling the counter current heat exchanger and one bottom section for the co-current heat exchange . Resulting in the following topology for a system with a total of 4 sections:

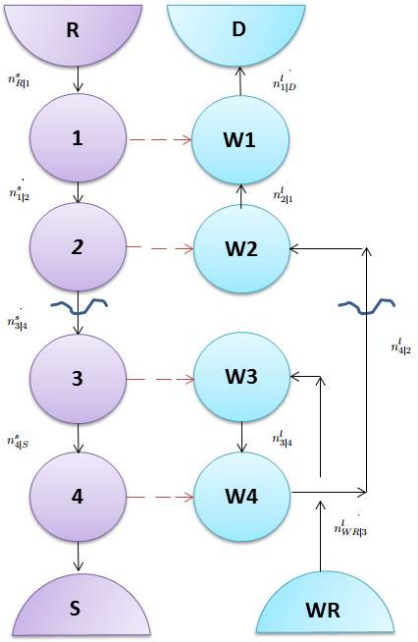


Figure 10: Topology of the bulk cooler model with a CSTR's in series approach

The initial temperatures of the interior stages of the bulk cooler must be guessed and the model will therefor take some time to achieve steady state. In the previous model, section 3.2, the cooling water was set to have one constant value throughout the simulation. It's safe to assume that this simplifications will be highly inaccurate for the CSTR in series system and will also set limitation for use of the model. Therefore an additional set of state variables for the liquid phase must be included.

3.3.1 Mass balance

Numbering the section from 1 to $n+2$ where the counter current heat exchanger is the sections from $i = 2$ til $i = n/2 + 1$ gives the following mass balance for ith section for the solid and the liquid phase in the counter current heat exchanger:

$$\frac{dN_i^s}{dt} = \hat{n}^s_{i-1|i} - \hat{n}^s_{i|i+1} \qquad for\ i = 1...(n/2+1) \tag{35}$$

$$\frac{dN_i^l}{dt} = \hat{n}^l_{i+1|i} - \hat{n}^l_{i|i-1} \qquad for\ i = 2...(n/2) \tag{36}$$

The central section separating the two heat exchangers:

$$\frac{dN_i^s}{dt} = \hat{n}_{i-1|i}^s - \hat{n}_{i|i+1}^s \qquad for\ i = (n/2+2) \tag{37}$$

The bottom heat exchanger section with co-current heat exchange:

$$\frac{dN_i^s}{dt} = \hat{n}_{i-1|i}^s - \hat{n}_{i|i+1}^s \qquad for\ i = (n/2+3)...n+2 \tag{38}$$

$$\frac{dN_i^l}{dt} = \hat{n}_{i-1|i}^l - \hat{n}_{i|i+1}^l \qquad for\ i = (n/2+4)...(n+2) \tag{39}$$

Unique liquid stages:

$$\frac{dN_{n/2+3}^l}{dt} = \hat{n}_{c,in|n/2+3}^l - \hat{n}_{n/2+3|n/2+4}^l \tag{40}$$

$$\frac{dN_{n/2}^l}{dt} = \hat{n}_{n+2|n/2}^l + \hat{n}_{n/2|n/2-1}^l \qquad for\ i = 2...(n/2) \tag{41}$$

The unique liquid stages are not of any importance for the mass balance as they will be further simplified below. However it's important to develop an understanding for the flow direction as it will be of high importance for the energy balance. The liquid states and the solid states for the interior sections can be further simplified as a result of the sections being situated directly on top of each other. Leaving no room for accumulation as incompressible flow is considered safe to assume for both the liquid and the solid phase in the range of the bulk coolers operating conditions. This in turn yields for the mass balances:

$$\frac{dN_i^s}{dt} = 0 \qquad for\ i \neq 1 \tag{42}$$

$$\frac{dN_i^l}{dt} = 0 \tag{43}$$

3.3.2 Energy balance

For the solid phase the energy balance bears close resemblance to a single CSTR model:

$$\frac{dU_i^s}{dt} = \hat{H}_{i-1|i}^s - \hat{H}_{i|i+1}^s - \hat{q}_{l_i|s_i} \qquad for\ i \neq 1, n/2+2 \tag{44}$$

By utilizing the procedure in 3.2 the energy balance can be transformed to:

$$N_i^s c_p(T_i)\frac{dT_i}{dt} = (h(T_{i-1}) - h(T_i))\hat{n}_{i-1|i}^s - (h(T_i) - h(T_i))\hat{n}_{i|i+1}^s - Q_{i|Wi} \tag{45}$$

$$\frac{dT_i^s}{dt} = \frac{1}{N_i^s a_{NPK1}}\left((T_{i-1} - T_i)a_{NPK1}\hat{n}_{i-1|i}^s - Q_{i|Wi}\right) \tag{46}$$

For the liquid phase the energy balance for the *ith* section for the counter current heat exchanger is highly similar:

$$\frac{dU_i^l}{dt} = \hat{H}_{i+1|i}^l - \hat{H}_{i|i-1}^l + \hat{q}_{l_i|s_i} \quad for\ i = 2...(n/2) \tag{47}$$

$$\frac{dT_i^l}{dt} = \frac{1}{N_i^l a_{H_2O1}} \Big((T_{i+1} - T_i) a_{H_2O1} \hat{n}_{i+1|i}^l + Q_{i|Wi} \Big) \tag{48}$$

For the co-current section:

$$\frac{dU_i^l}{dt} = \hat{H}_{i-1|i}^l - \hat{H}_{i|i+1}^l + \hat{q}_{l_i|s_i} \quad for\ i = (n/2+4) + (n+2) \tag{49}$$

$$\frac{dT_i^l}{dt} = \frac{1}{N_i^l a_{H_2O1}} \Big((T_{i+1} - T_i) a_{H_2O1} \hat{n}_{i+1|i}^l + Q_{i|Wi} \Big) \tag{50}$$

Resulting in a system with $1 + n \cdot 2$ state variables

3.3.3 State identification

For the CSTR in series model the states will be the mass in section one for the solid phase and the temperature for the *ith* stage both for the liquid and the solid. In addition there is extra heat balances for the solid a the center section and the top section where there is no heat transfer between the liquid and the solid.

$$\underline{x} = \begin{bmatrix} N_1^s & T_1^s ... & T_n^s & T_1^l ... & T_n^l \end{bmatrix}^T \tag{51}$$

$$\underline{u} = \begin{bmatrix} \hat{n}_{R|BC}^s & \hat{n}_{BC|S}^s & T_R & \hat{n}_{D|n}^l & \hat{n}_{1|WR}^l \end{bmatrix}^T \tag{52}$$

$$\underline{\gamma} = \begin{bmatrix} (T_{c|in} & T_{c|out}) \end{bmatrix}^T \tag{53}$$

$$\underline{\Theta} = \begin{bmatrix} a_{NPK,1} & UA \end{bmatrix}^T \tag{54}$$

3.3.4 Simulink model of the CSTR in series

The Simulink model is quite similar to the previous model and once again the differential equations was solved by use of the S-function block:

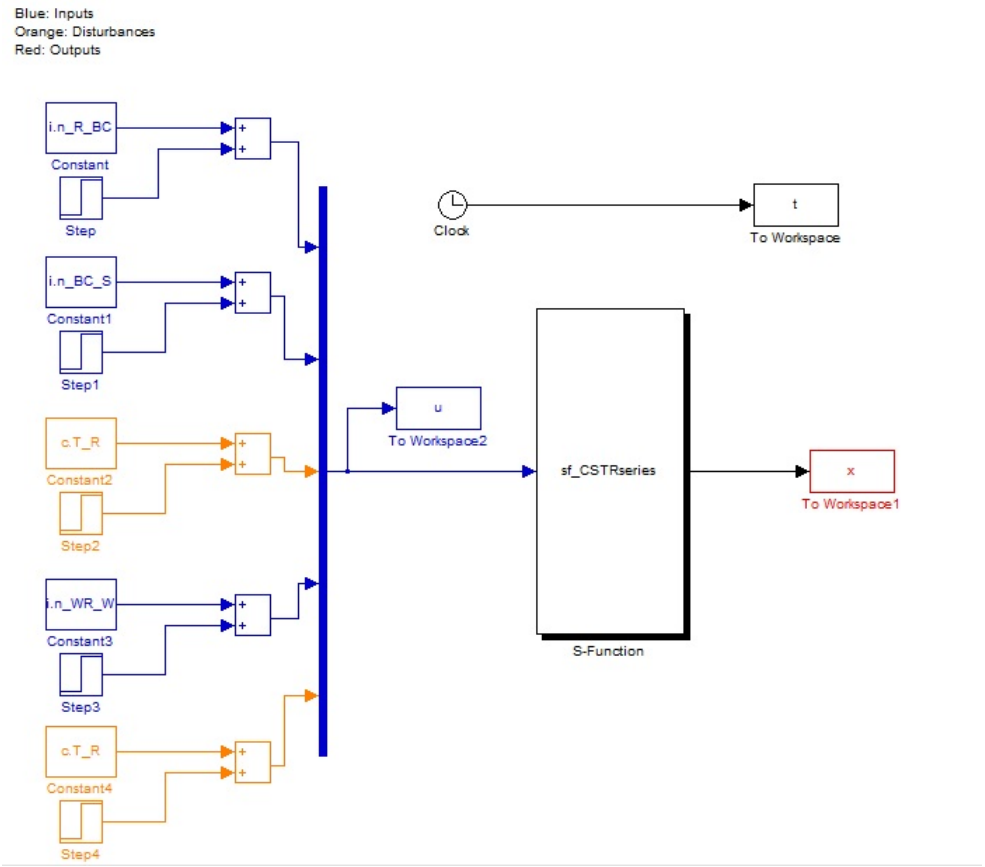


Figure 11: Simulink model of the "CSTR in series" model

The S-function block code can be seen in A.5.3. When inserting the initial conditions used in designing the bulk cooler;

$$T^l_{WR} = 15°C \qquad T^s_R = 43°C \tag{55}$$

$$\hat{n}^l_{WR|n/2} = 50ton/hC \qquad \hat{n}^s_{n|s} = 210ton/h \tag{56}$$

$$\tag{57}$$

The model returns the following temperature profile for the steady state system:

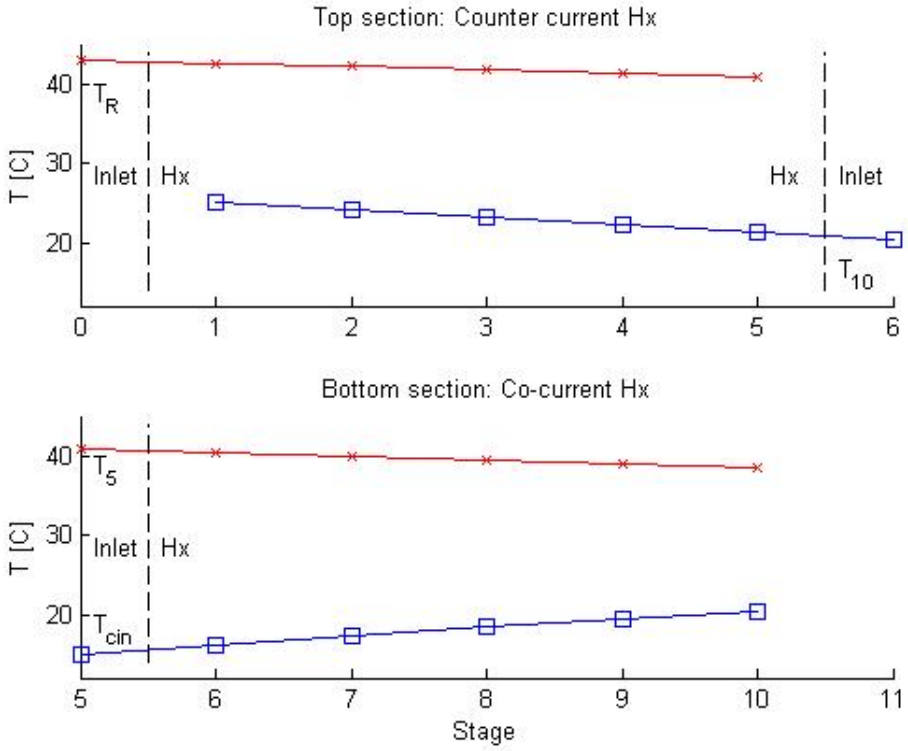


Figure 12: Steady state temperature profile with design data as input

The steady state system in figure 12 has the following outlet temperatures:

$$T^l_{WR} = 24.52°C \qquad T^s_R = 38.56°C \tag{58}$$

4 Experimental

4.1 MATLAB model validation

4.1.1 Comparison of design data and model data

The design data provided in 31 operates with the following temperature and flow data:

$$T^l_{WR} = 15°C \qquad T^s_R = 43°C \tag{59}$$

$$T^l_D = 30°C \qquad T^s_S = 32°C \tag{60}$$

$$\hat{n}^l_{WR|n/2} = 50ton/hC \qquad \hat{n}^s_{n|s} = 210ton/h \tag{61}$$

$$\tag{62}$$

Comparing these design values to the values acquired by the model in 58 initially reveals that the there is a clear mismatch between the calculated heat coefficient (U) and the design heat coefficient. This is shown in the temperature profile as both streams can not reach their target values. To try to match the design values a scaling parameter is introduced. In the MATLAB script A.5.2 the parameter $p.alfa$ is inserted. This factor multiplies the current U value an therefore amplifies or decreases the heat transfer. Setting this value to 5.8 yields the following temperature profile:

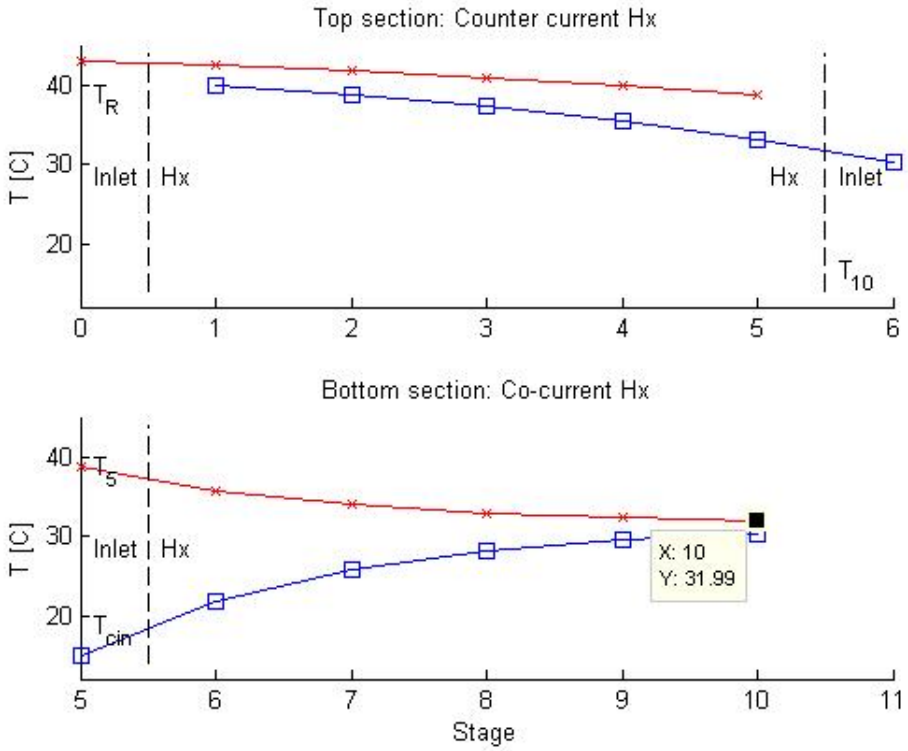


Figure 13: Steady state temperature profile with design data as input with adjusted U

The fitted steady state system in figure 13 has the following outlet temperatures:

$$T^l_{WR} = 39.98°C \qquad T^s_R = 31.99\circ C \tag{63}$$

With these changes the outlet temperature of the cooling water severely overshoots the design temperature. To check that the energy transfer in the model is not flawed, a total steady state mass balance are calculated with the physical properties used in the model:

$$0 = (T_R - T_S)a_{NPK1}\hat{n}^s_{n|S} - Q \tag{64}$$

$$0 = (T_{WR} - T_D)a_{H2O1}\hat{n}^l_{WR|D} + Q \tag{65}$$

$$(T_R - T_S)a_{NPK1}\hat{n}^s_{n|S} = -(T_{WR} - T_D)a_{H2O1}\hat{n}^l_{WR|D} \tag{66}$$

$$1445309.2966J/s = -1445580.9468J/s \tag{67}$$

$$\tag{68}$$

Which tells us that the system has in fact reached steady state and that there is much more likely that the current displacement from design values are caused by mismatched heat coefficients. Considering that the physical properties of water repeatedly has been validated over the years it is most likely that it is the heat capacity of the NPK that is flawed. Also, the existing data only comprehends high N products and the specific heat capacity of KCl is only $690J/kgK$ [7] which is considerably less than the value calculated in 8. It is therefore safe to assume that the heat capacity used in designing the cooler is considerably less. To further fit the model against design values a new parameter is introduced $p.beta$. This value is multiplied to the heat capacity extracted from the provided enthalpy data and modified until the output temperature of the cooling water meets the design temperature while simultaneously re-fitting $p.alfa$. The following temperature profile is achieved:

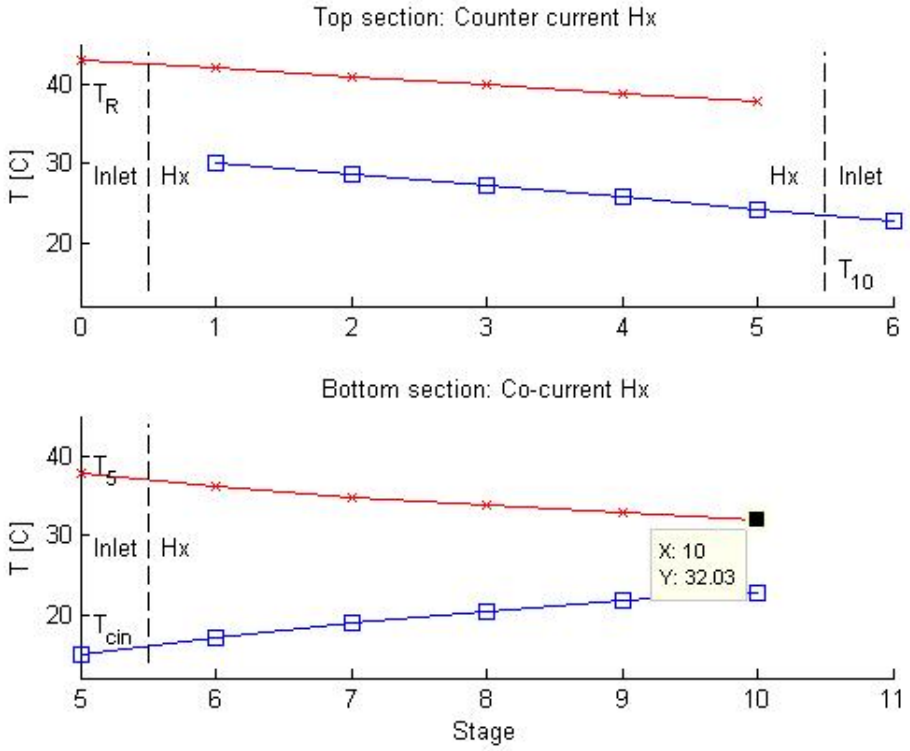


Figure 14: Steady state temperature profile with design data as input

This model has the following outlet temperatures and scaling parameters:

$$T^l_{WR} = 29.93°C \qquad T^s_R = 32.03°C \tag{69}$$

$$p.alfa = 2.22°C \qquad p.beta = 0.6°C \tag{70}$$

And the total mass balance:

$$(T_R - T_S)a_{NPK1}\hat{n}^s_{n|S} = -(T_{WR} - T_D)a_{H2O1}\hat{n}^l_{WR|D} \tag{71}$$

$$864272.3668J/s \approx 864108.9127J/J/s \tag{72}$$

4.2 Modelling the phase change

To further improve the model the NPK heat coefficient is improved by use of the technique explained further in 3.1.3. Basically, the phase change is modelled as a linear function over the span of $1°C$. With using the previous set of scaling parameters (section 4.1.1):

$$p.alfa = 2.22°C \qquad p.beta = 0.6°C \tag{73}$$

The model now gets the following temperature development:

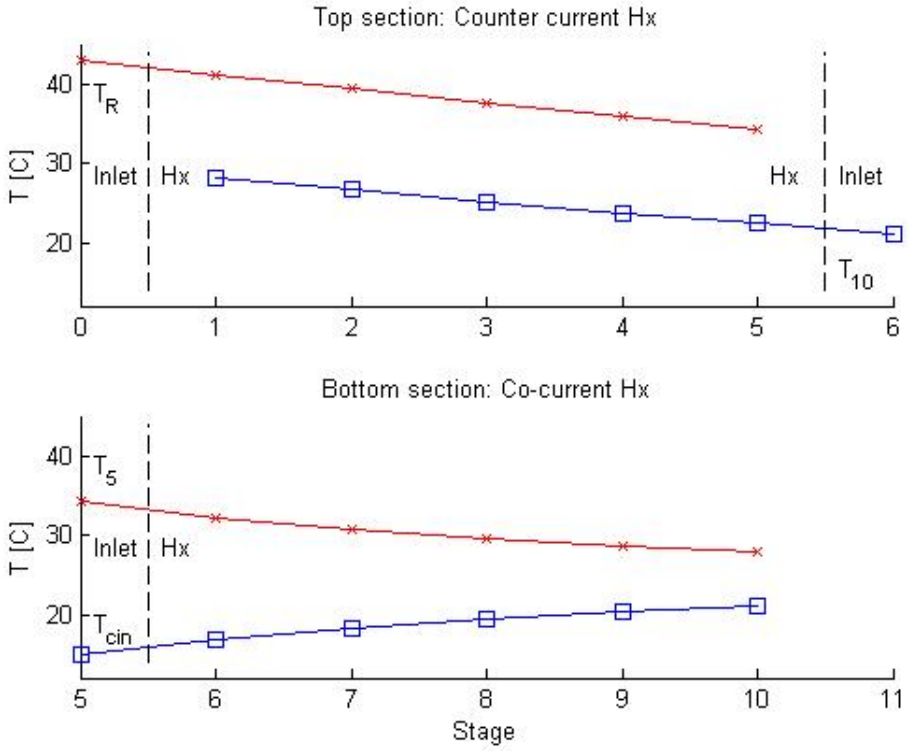


Figure 15: Steady state temperature profile of the bulk cooler with the phase change modelled over the span of $1°C$.

The outlet temperatures are:

$$T^l_{WR} = 28.11°C \qquad\qquad T^s_R = 27.84°C \tag{74}$$

Changing the cp model has led to an overall sever performance increase. The NPK is cooled an additional $4.19°C$ while the overall energy transfer from the liquid phase goes down with $40921J/s$. This allows for readjusting of the scaling parameter. Setting $p.beta = 0.68$ yields:

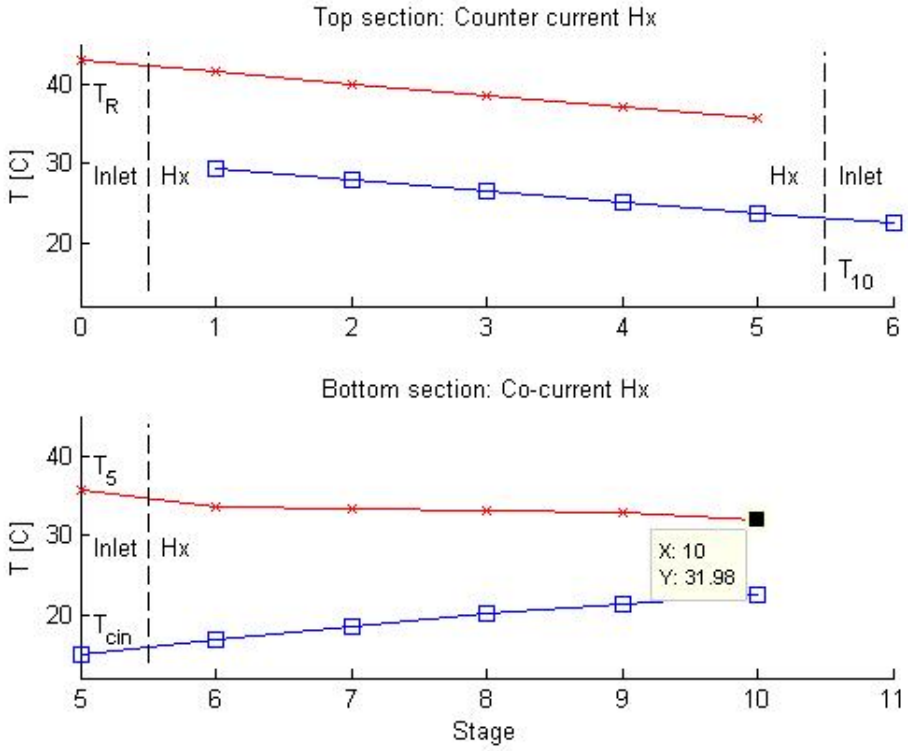


Figure 16: Adjusted steady state temperature profile of the bulk cooler with the phase change modelled over the span of $1°C$.

The new outlet temperatures are:

$$T^l_{WR} = 29.22°C \qquad\qquad T^s_R = 31.98°C \tag{75}$$

Here the phase change becomes much more transparent and the model returns that the phase change happens between stage 6 and 9.

4.3 Single counter current model comparison

The chosen set up for the bulk cooler is quite interesting, to further analyse the counter current - co-curent set up a additional simulation was performed. A new model was developed, changing the current model to comprehend one single counter current heat exchanger with the same overall heat transfer area and physical properties as in section 4.2. The model uses the parameters fitted in section 4.1.1.

$$p.alfa = 2.22°C \qquad\qquad p.beta = 0.6°C \tag{76}$$

The simulation yields the following temperature profile:

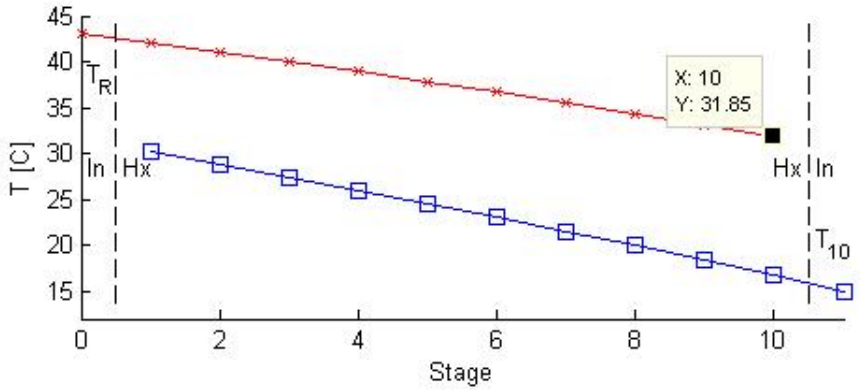


Figure 17: Steady state temperature profile of a single counter current heat exchanger with design data as input

The outlet values are:

$$T^l_{WR} = 29.93°C \qquad T^s_R = 32.03°C \tag{77}$$

With the total heat balance:

$$(T_R - T_S)a_{NPK1}\hat{n}^s_{n|S} = -(T_{WR} - T_D)a_{H2O1}\hat{n}^l_{WR|D} \tag{78}$$

$$877783.7171J/s = 878105.9564J/s \tag{79}$$

$$\tag{80}$$

A difference of $13754J/s$ from the corresponding heat balance in 72 and translates to only $0.17°C$ further cooling of the NPK at the current rate of $210t/h$.

The counter current model is then checked with use of the phase change model in 4.2 ($p.beta$ raised to 0.68):

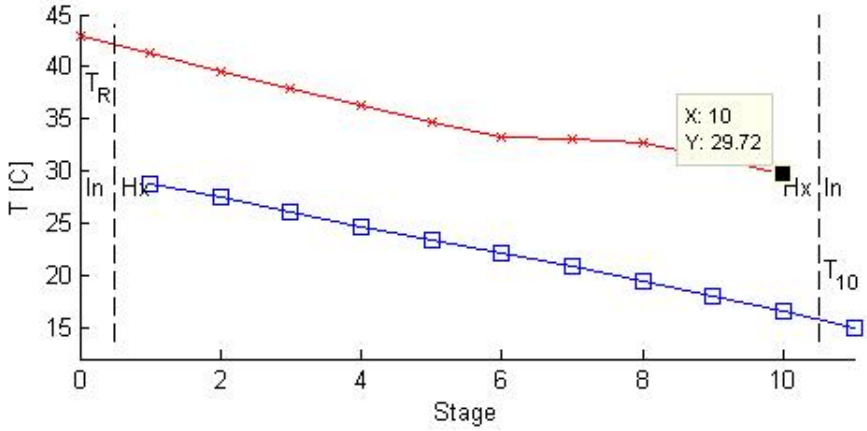


Figure 18: Steady state temperature profile of a single counter current heat exchanger with design data as input

The outlet temperatures:

$$T^l_{WR} = 28.83°C \qquad T^s_R = 29.72°C \tag{81}$$

Revealing a much larger ΔT for the NPK. Which reveals the impact of the smaller heat coefficients outside of the range of the phase change.

4.4 Controlling the inlet temperature with cascade control

To make the model represent the real system dynamically, a simplified temperature control is introduced. A PI controller will adjust the flow rate of the cooling water in order to achieve the desired outlet temperature for the NPK - $T^s_{BC|S} = 32$. This changes the Simulink model to the following:

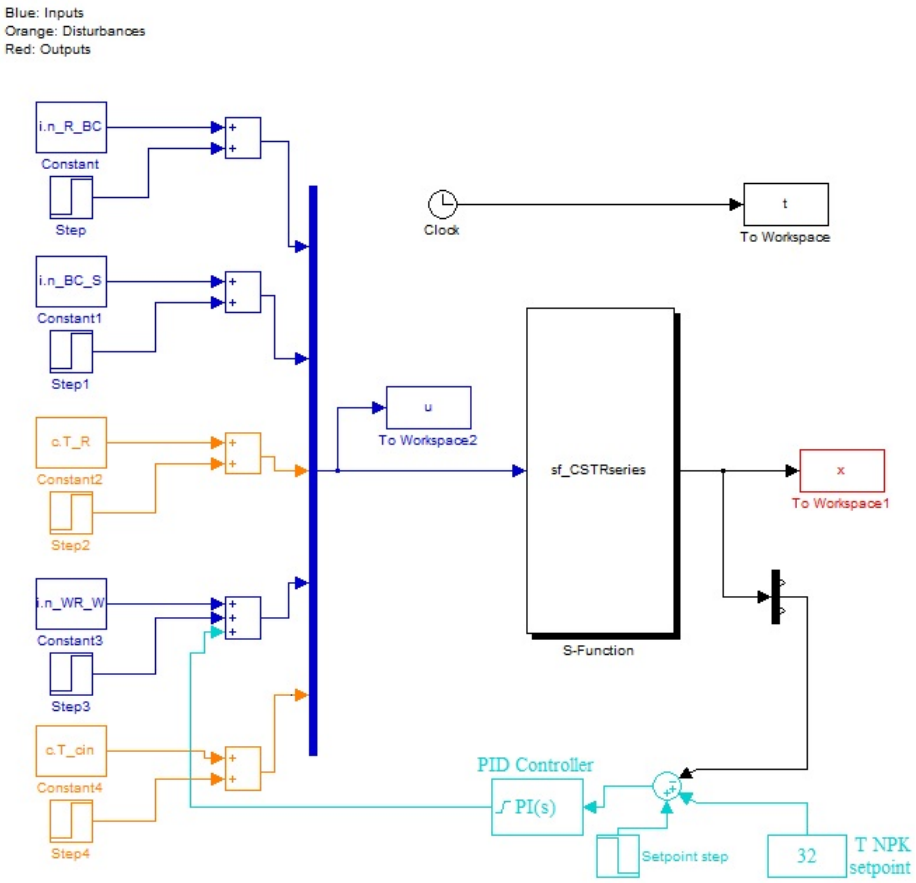


Figure 19: Simulink model of the "CSTR in series" model with PI temperature control on the outlet temperature of the NPK

To tune the controller Skogestad's SIMC rules [3] for PI control is used:

$$K_C = \frac{1}{k}\frac{\tau_1}{\tau_c + \theta} \tag{82}$$

$$\tau_I = min\tau_1, 4(\tau_c + \omega) \tag{83}$$

Where the needed information of the model is:

- Plant gain, k
- Dominant lag time constant, τ_1
- Time delay, θ

An estimate for the factors in the SIMC rules can be obtained by an open loop step response. This was performed by letting the bulk cooler reach

steady state and then taking a 10% step on the inlet mass flow of cooling water. In the Simulink model the step change is performed by the block Step3, which adds a 10% to the constant block $i.n_W R_W$. The figure below shows the resulting step response for the outlet temperature:

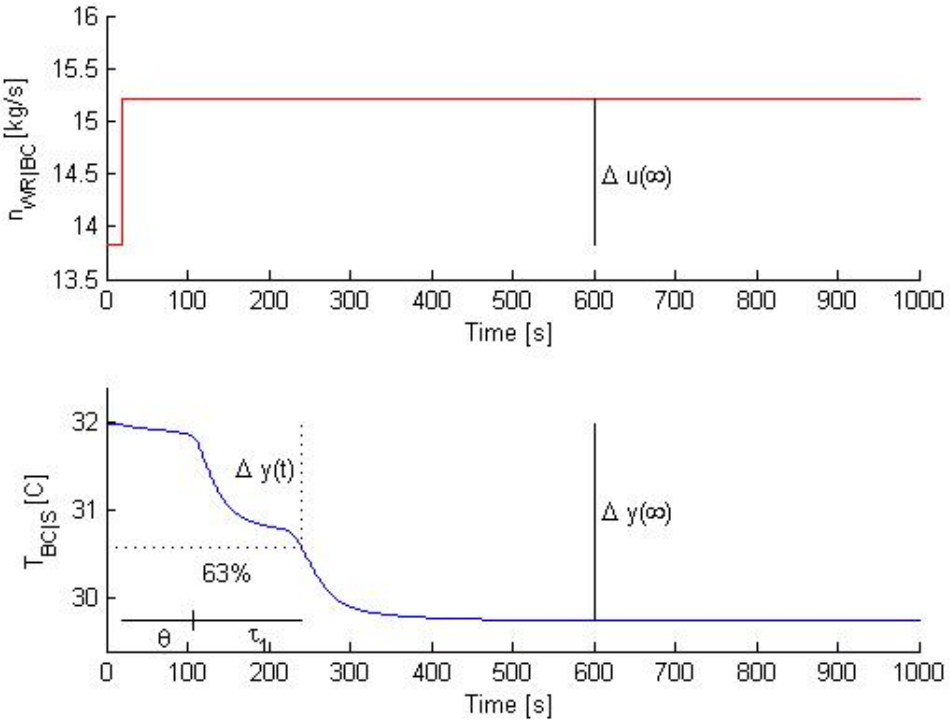


Figure 20: Step response of the dynamic bulk cooler model

Which shows a response hard to classify. The nature of the double heat exchanger seems to be transparent when looking at the s-shaped response. For the first tuning attempt the entire response is approximated as a first order step response and the SIMC parameters are calculated as shown in figure 20 and their values are:

$$k = \frac{\Delta y(\infty}{\Delta u(\infty)} = \frac{-2.2272}{1.3835} = -1.6098 \tag{84}$$

$$\tau_1 = 133.4 \tag{85}$$

$$\theta = 86.8 \tag{86}$$

Which yields:

$$K_C = \frac{1}{-1.6098}\frac{133.4}{2 \cdot 86.8} = -0.5179 \tag{87}$$

$$\tau_I = \tau_1 \tag{88}$$

Testing the controller by introducing a disturbance. The inlet temperature of NPK are increased by 1% and the controllers response is portrayed in figure 21:

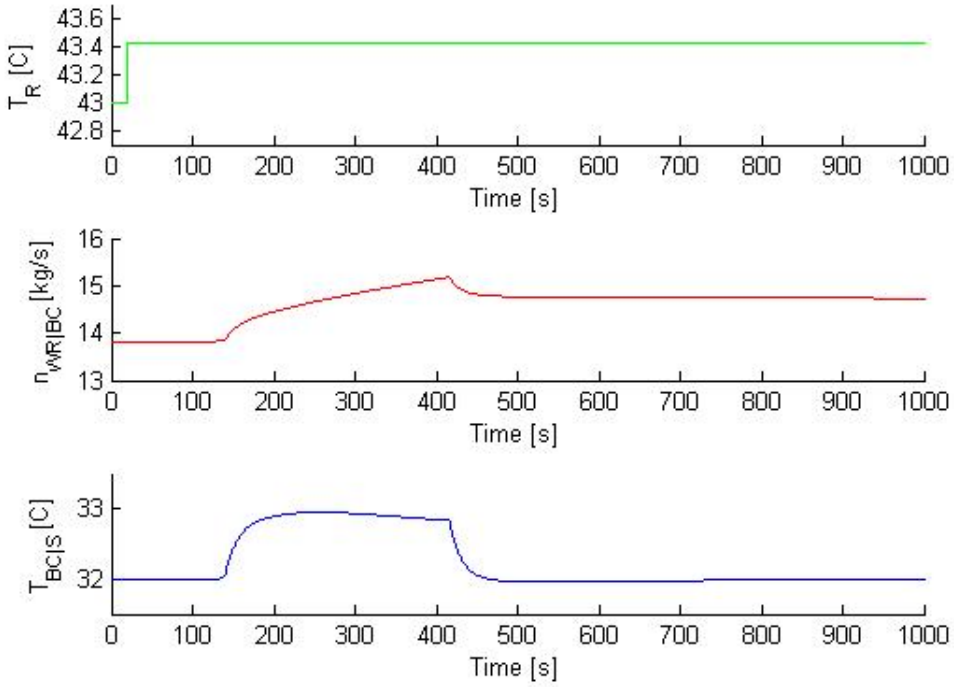


Figure 21: Shows the controller performance of a PI controller with the cooling water flow as the manipulated variable with $K_c = -0.5179$ and $\tau_I = 133.4$ exposed to a 1% increase in the inlet NPK temperature

The response of the system is somewhat slow, which is expected for a system with a considerable time delay. In an attempt to improve the controller, the approximation of the SIMC rules where performed over a shorter time frame in order to increase the controller response time

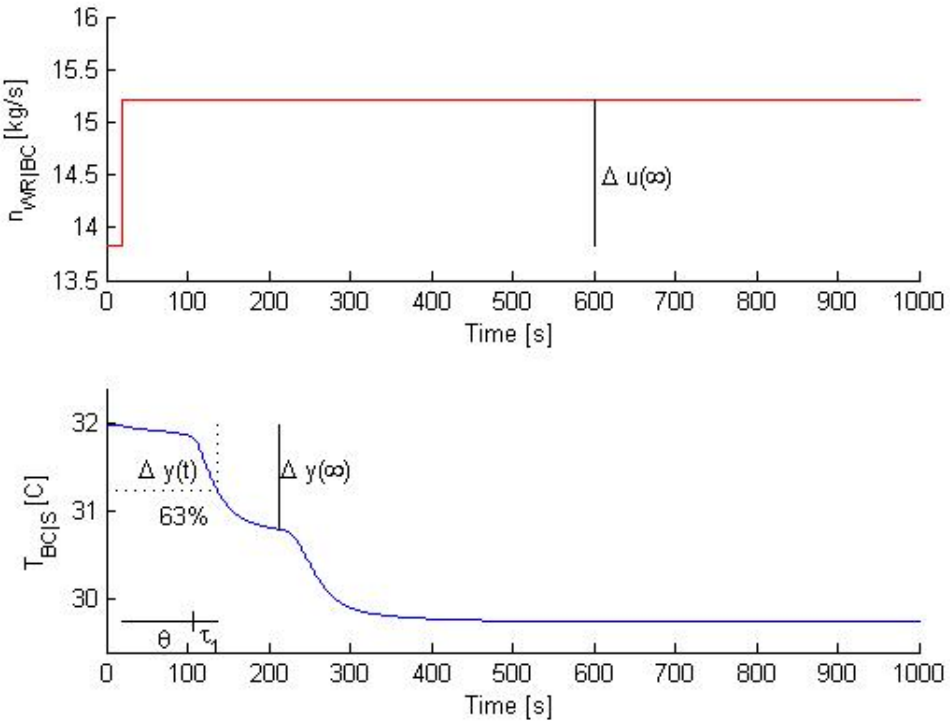


Figure 22: Step response of the dynamic bulk cooler model where "$y(\infty)$" is considered over a smaller time frame. Ignoring any response beyond

Calculating the new controller parameters by using the SIMC rules

$$k = \frac{\Delta y(\infty}{\Delta u(\infty)} = \frac{- - 1.13}{1.3835} = -0.8529 \tag{89}$$

$$\tau_1 = 30 \tag{90}$$

$$\theta = 86.8 \tag{91}$$

$$K_C = \frac{1}{-0.8529} \frac{30}{2 \cdot 86.8} = -0.2198 \tag{92}$$

$$\tau_I = \tau_1 \tag{93}$$

By using the same disturbance as in 21 the controller response is portrayed in figure 23

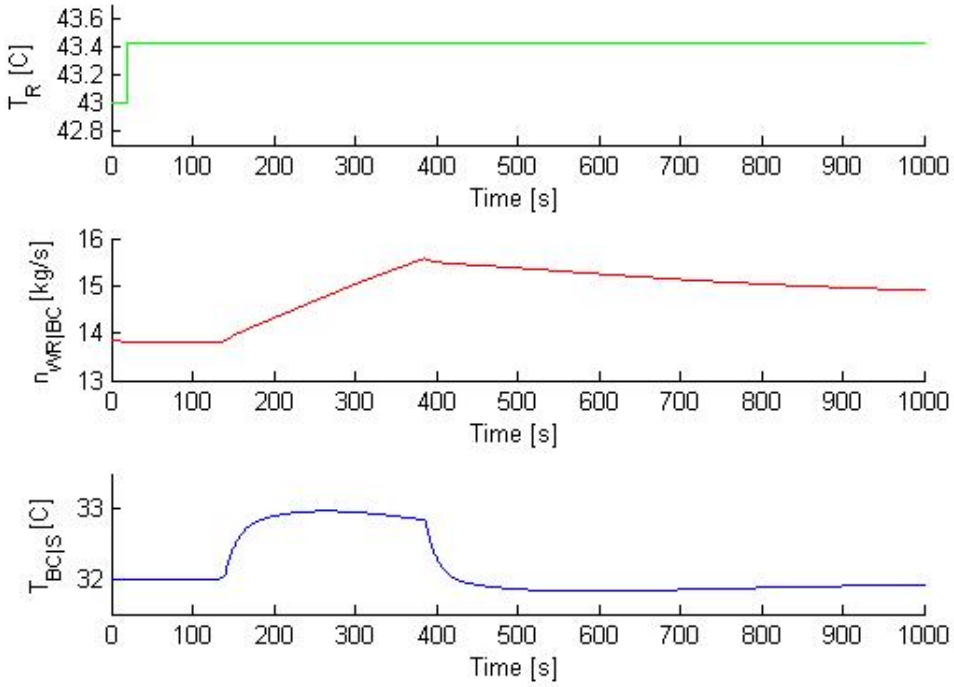


Figure 23: Controller performance of a PI controller with the cooling water flow as the manipulated variable and $K_c = -0.2198$ and $\tau_I = 30$ exposed to a 1% increase in the inlet NPK temperature

Figure 23 shows a slightly faster response to the disturbance and the new controller gain and integral time will be kept for the remainder of the simulations. To further validate the controller a set point step test was performed at $t = 500s$. Figure 24 shows the performance:

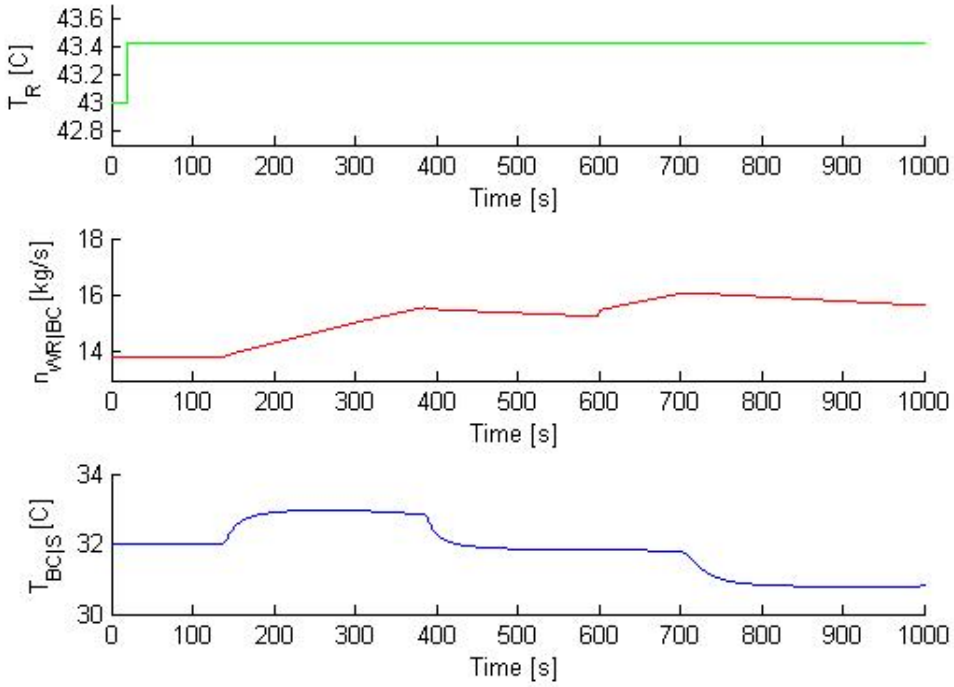


Figure 24: Controller performance of a PI controller with the cooling water flow as the manipulated variable and $K_c = -0.2198$ and $\tau_I = 30$ exposed to a 1% increase in the inlet NPK temperature at $t = 20s$ and a $-1°C$ set point change at $t = 600s$

4.5 The $250ton/hr$ scenario

The main future goal for the NPK factory is to achieve a production rate of $250ton/hr$. By using the improved temperature controlled CSTR in series model, the goal is to simulate the required amount of cooling water needed to achieve the required cooling. This value will then be compared to the technical evaluation performed by Yara in B.4.1. The technical evaluation operates with the following conditions:

$$\hat{n}^s_{R|BC} = 250ton/hr \qquad T^s_R = 47.6°C \qquad T^s_{BC|S} = 30°C \tag{94}$$

Inserting the parameters in the model developed in section 4.4 yields:

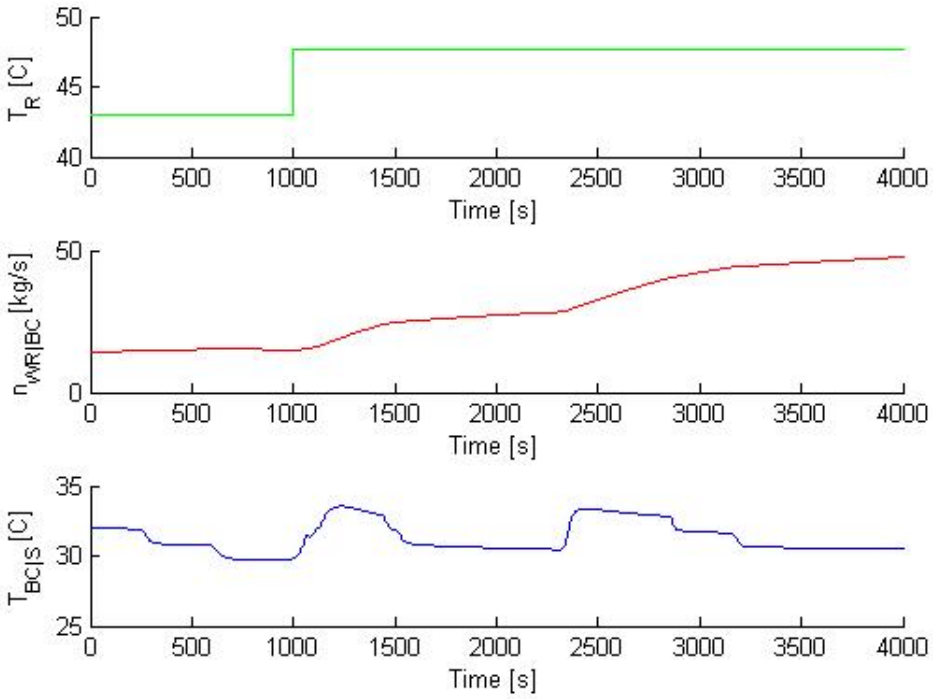


Figure 25: Model converging on a steady state with $250 ton/hr$ NPK

At the end of the simulation the water flow approaches $50 kg/s$. Which translates to $180.7 m^3/h$ which is much higher than the $136 m^3/h$ proposed in section B.4.1. This in turn can be seen in relation to the temperature profile in the end of the horizon:

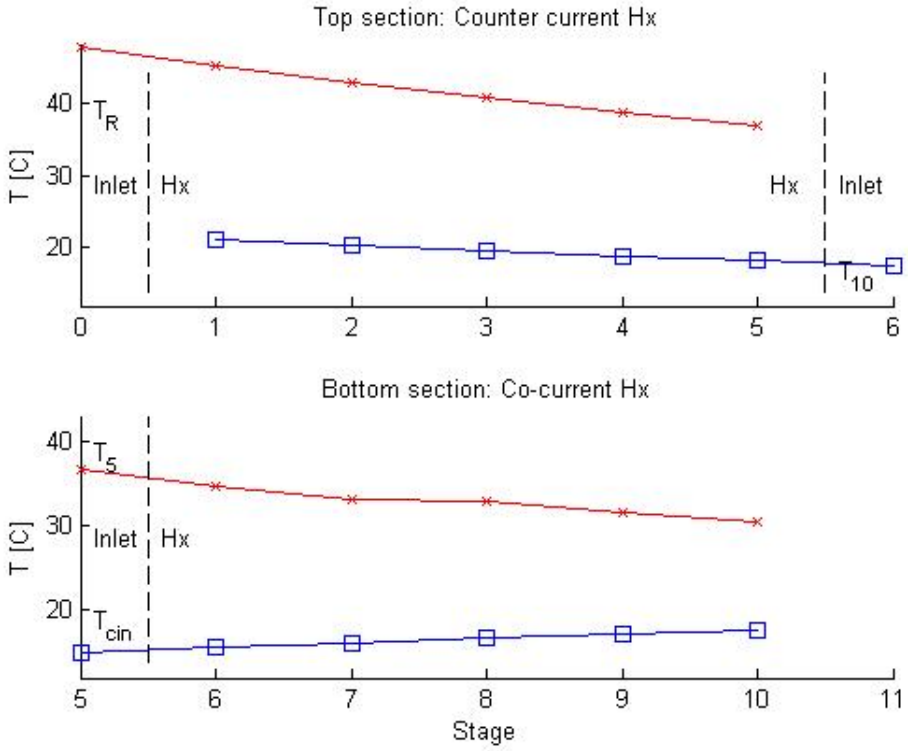


Figure 26: Temperature profile for the end of the horizon of 25

Here the T_D^l is only $21.08°C$, which implies that the U calculation and scaling with $p.alfa$ is insufficient for high production rates.

4.6 Comparison of tuned model and process data

On the 11th of September the NPK4 factory manufactured an order of NPK type $16-16-16(MOP)$ where the K source is primarily Potassium Chloride. The trends reveals the nature of the unsteady NPK process. Operating the factory yields by default some instability, partly because of a technique known as "bambling" which is indispensable for long term operation (long term as in more than 1 hour). "Bambling" is a local term used to describe the process of stopping the flow of salts to raise the temperature in the centrifuge and mixer, and in turn melt down solidified particles on mentioned units. The high temperature leads to a popcorn like product as the NPK fertilizer has a temperature that is too high for successful prilling, causing the particles to stick to each other when hitting the bottom of the tower. This will in turn lead to increased mass flow over the crusher and a temporary accumulation of NPK in the sieving/crushing system. The result, as seen on the trends in figure 33, is notable fluctuations on the flow of NPK making it hard to locate

a steady state system to compare with the model. In addition this leads to fluctuations in the potassium chloride content and therefore changing the physical properties of the NPK. The first process state that will be compared to the model is the state at $03:29:51$ where the fluctuations are somewhat in a median state and close to the design value. The process values are:

$$T^l_{WR} = 18.8°C \qquad T^s_R = 42.5°C \tag{95}$$

$$T^l_D = 33.1°C \qquad T^s_S = 32.3°C \tag{96}$$

$$\hat{n}^l_{WR|n/2} = 58.6ton/hC \qquad \hat{n}^s_{n|s} = 209.9ton/h \tag{97}$$

Entering the inlet values in the model yields:

$$T^l_D = 30.21°C \qquad T^s_S = 31.1°C \tag{98}$$

Introducing the error:

$$\epsilon^l = T^l_{D.model} - T^l_{D.trend} \qquad \epsilon^s = T^s_{S.model} - T^s_{S.trend} \tag{99}$$

$$\epsilon^l = -2.89 \qquad \epsilon^s = -1.2 \tag{100}$$

To compare, data in the same time range with similar flow is used ($03:19:11$):

$$T^l_{WR} = 18.3°C \qquad T^s_R = 41.9°C \tag{101}$$

$$T^l_D = 33.1°C \qquad T^s_S = 31.8°C \tag{102}$$

$$\hat{n}^l_{WR|n/2} = 56.7ton/hC \qquad \hat{n}^s_{n|s} = 211.3ton/h \tag{103}$$

The model returns

$$T^l_D = 29.51°C \qquad T^s_S = 28.99°C \tag{104}$$

$$\epsilon^l = -3.59 \qquad \epsilon^s = -2.81 \tag{105}$$

Which is an alarmingly high increase in the error for systems only 10 minutes a part and with similar states. Looking at the temperature profile reveals an interesting limitation of the model:

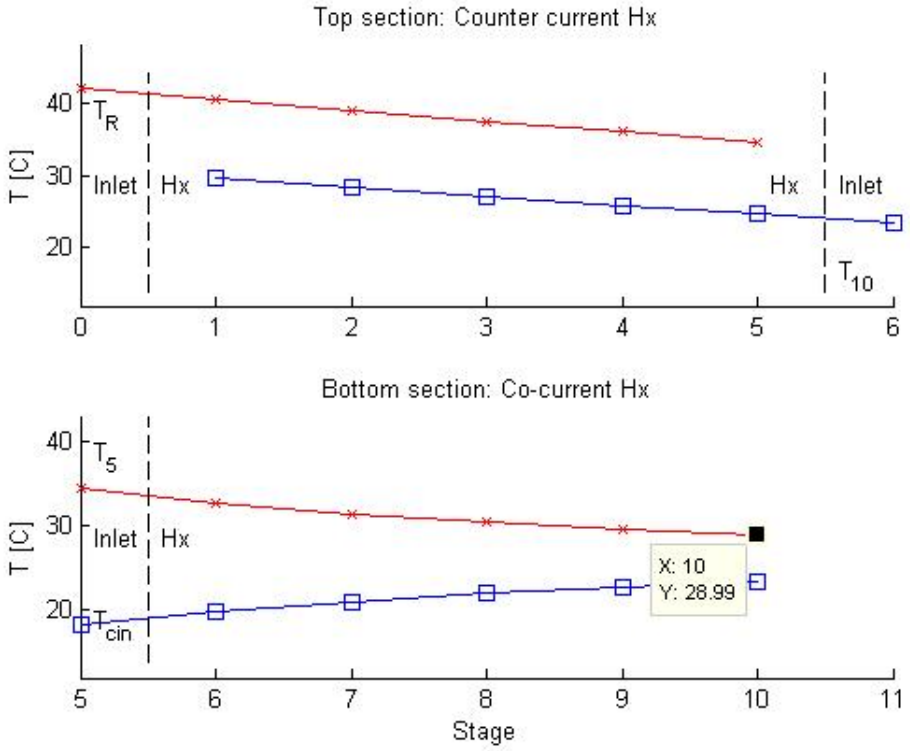


Figure 27: Temperature profile for the model with inlet parameters corresponding to the values in 33 at03 : 19 : 11

The phase change is no longer visible in the temperature profile. The phase change operates in the range between $33 - 34°C$ and since the temperature difference between stage 5 and stage 6 for the solid lies outside that range the phase change heat coefficient never gets called in the model. Resulting in a net low heat coefficient for the NPK and a low outlet temperature.

In order of conditioning the model to minimize the error the trend data in itself must be validated. With a simple MATLAB script, see section A.8, the mean heat transfer coefficient for the NPK was calculated by assuming that the water measurements are correct. Which in turn is assumed fair compared to the measurements of NPK.

$$\frac{Q_water}{\Delta T^s_{NPK} \hat{n}^s_{NPK}} = cp^s_{NPK} \tag{106}$$

By executing the run file the Cp calculations for November 11th fluctuate between $1580.7 J/s$ and $1896 J/s$ which can partially be explained by the system not being in steady state but also strongly indicate that the measurements

are not completely accurate. With an unconfirmed heat coefficient (scaled by a parameter) and with the unsteady process, further model improvement is considered redundant at this point. Laboratory test is recommended for further improvement.

5 Discussion

5.1 Modelling the phase change

For the section of modelling the phase change,3.1 with figure 3, shows the superiority of choosing to model the phase change over a span of $\Delta T = 1°C$. This is further confirmed by the possibility of relaxing the fitting parameters $p.alfa$ and $p.beta$ when using the method. The weakness of the heat coefficient lies mostly within the limited data for different types of NPK. The phase change enthalpy will change with the fraction of ammonium nitrate present, which in turn relies on the current NPK specification. Also, by studying section B.1 and B.2 it is transparent that the temperature which the phase change occurs is dependent on the product composition. To improve the phase change model laboratory experiments studying the specific heat coefficient for the range of NPK products is recommended

5.2 The fitting parameters

The parameters used for fitting needs some justification. First of all the calculation of U relies on temperature difference of the solid and the liquid. The calculations are made by averaging the temperature for the solid and liquid and then use the difference as the driving force of the heat transfer. Considering the process faces a co-current section and a phase change that simplification is rather weak. This is also disclosed by the model as the driving force varies considerably in the bottom section. In addition, the model uses the entirety of the heat transfer area while the hold up has been modified with the void factor of packed spheres (see the parameters script in the MATLAB section). These voids does not lead to any adjustment in modelling the U. These factors combines to a justification of implementing $p.alfa$ as a $p.alfa > 1$ reduces the impact of the area when calculating U. Further development in this area can be focused to adjusting for heat conduction to the air pockets and heat conduction to the solid spheres.

$p.beta$ adjusts the specific heat capacity of the fertilizer and is in section 4.1.1 set to 0.68. The decreased heat capacity can be seen in relation to the impact of having more salts (f.ex KCl) which has a heat capacity of significantly lower the ammonium base compounds (see section **??**). This justifies decreasing the heat capacity by using $p.beta$ as $cp_{KCl} = 695$ [7] seems severely less than calculated heat capacity of the high N product using the linear approach in section 3.1.1.

5.3 The achieved temperature profiles

In general the temperature profiles returned after adjusting with the fitting parameters seems highly viable. The characteristic profiles of both the co-

current heat exchanger and the counter current is easily identified in the models. Also, when implementing the phase change the model responds with the expected flat temperature profile in the area of the phase change. However, some problems were experienced when dealing with strong disturbances. If the energy balances of two adjacent sections starts to simultaneously stabilize in a temperature range higher and lower than the phase change, the model may in some cases skip the phase change completely. This is an effect of the logic operator initiating the high specific heat capacity connected to the phase change, never getting initiated if none of the sections operates in the specified range of the phase change. This in turn leads to a sudden considerable jump in the ΔT for the solid yielding a simulation which don't correlate with the real system. Suggested further development in this is introducing a flag catching the jump over the phase change. A quick fix would be to increase the number of stages, but this in turn directly affect the computational cost of the model.

5.4 The dynamic simulations

The dynamic simulations showed promising results in the range of $210 ton/h$. When trying to move up to $250 ton/h$ the temperature profile of the cooling water is quite flat while the flow rates are quite high. It seems like the heat transfer is to slow in comparison to the flow controller. This implies that the parameters $p.alfa$ and $p.bravo$ is insufficient in describing the systems stepping largely enough away from the design specifications. A suggested soution to this problem is changing the parameters from constant to linear functions scaling with the flow rate of NPK.

6 Conclusion

By studying the temperature profile of the adjusted model in figure **??**, it is safe to assume that the general strategy of modelling the bulk cooler as several CSTR units in series is sufficient. All the expected characteristics are visible and present. The model displays the phase change and the characteristic temperature profile for both the co-current and counter current heat exchanger. Which also means that the selected specific heat capacity approximation (modelling the phase change over a span of $\Delta T = 1$) has been a successful strategy.

The parameters $p.alfa$ and $p.beta$ does not have sufficient backing for claiming viability outside of the $210 ton/hr$ design rate. As portrayed figure 26 shows how the increasing flow rate of water is faster than the internal transfer. Yielding a model where increasing the flow rate has a less and less effect on the temperature of the NPK. This suggests that the $p.alfa$ parameter needs readjusting when entering high flow rate territories

References

[1] Preisig, H., "TKP4135 Process systems engineering script", *https://dl.dropboxusercontent.com/u/19261469/PSE_script_03.pdf*,

[2] Jakobsen, H., "Fixed Bed Reactors", *http://www.nt.ntnu.no/users/jakobsen/TKP4145/fixedbed_2011.pdf*,

[3] Skogestad, S. Grimholt, C., "The SIMC method for smooth PID controller tuning", *http://www.nt.ntnu.no/users/skoge/publications/2012/skogestad-improved-simc-pid/old-submitted/simcpid.pdf*,

[4] National institute of standards and technology, "Water", *http://webbook.nist.gov/cgi/cbook.cgi?ID=C7732185&Type=JANAFL&Plot=on*,

[5] Solex thermal, Bulk cooler model, *http://www.solexthermal.com/products/cooling/cooling-using-water-products-to-400c*,

[6] Polynomial fitting, *http://www.originlab.com/www/helponline/origin/en/UserGuide/Polynomial_Regression_Results.html*,

[7] Data sheet KCl, janis. *http://www.janis.com/Libraries/Window_Transmissions/PotassiumChloride_KCl_TransmissionCurveDataSheet.sflb.ashx*,

7 Variable List

7.1 Indexes and special nomenclature

Index	Description
α	phase index
i	Species index
j	stage index
$\hat{x}$	Rate of unit x
Δ	Net change in unit

7.2 Variables

Variable	Units	Description
A	m^2	Area
a_i	$J/kg°C$	Specific heat parameter
BC	-	Bulk Cooler
Cp_i^α	$J/kg°C$	Specific heat coefficient
H_i^α	J/kg	Enthalpy
h_i^α	J/kg	Specific enthalpy
K	J/kg	Enthalpy constant
k	-	Plant gain
N_j^α	kg	Mass
$\hat{n}_{j-1\|j}^\alpha$	kg/s	Rate mass transport, from j-1 to j
p	Pa	Pressure
R	-	NPK reservoir, inlet conditions
S	-	NPK storage reservoir
T_j^α	C	Temperature
U	J/kg	Internal energy
WR	-	Water reservoir, inlet conditions
τ	s	lag time constant
θ	s	Time delay

A MATLAB-code

A.1 Model with simplified lumped system (CSTR approach)

The system i solved by use of Simulink, see figure 6

A.1.1 Run file

```
%% Published by Bjorn Tore Mathisen
% Model of the Bulk Cooler in NPK 4 as part of a specialization
% subject H2013.
% The following model approximates the Bulk Cooler as a CSTR:

clc
clf
clear all
close all

% Process constants:
Siconstant

% Global parameters
Siparameters

%%
% Disturbances
 time_n_R_BC = 20;         % Steptime [min]
 step_n_R_BC = 0.1;          % Stepfac from init value
 time_n_BC_S = 20;         % Steptime [min]
 step_n_BC_S = 0.0;          % Stepfac from init value
 time_T_R      = 20;
 step_T_R      = 0.0;

%Starting simulink model
sim('CSTRmod.mdl')

figure(1)
subplot(311)
hold on
%axis([0 40 -4*10^5 -3.5*10^5])
plot(t,u(:,1),'-r')
ylabel( 'n_{R|BC} [kg/min]')
xlabel('Time [s]')

figure(1)
subplot(312)
hold on
%axis([0 40 -4*10^5 -3.5*10^5])
plot(t,x(:,1),'-b')
ylabel('N_{BC} [kg]')
xlabel('Time [s]')
```

```
44
45 figure(1)
46 subplot(313)
47 hold on
48 axis([0 1000 30 33])
49 plot(t,x(:,2),'-b')
50 ylabel('T_{BC} [C]')
51 xlabel('Time [s]')
52
53 figure(2)
54 subplot(311)
55 hold on
56 %axis([0 1000 42 48])
57 plot(t,u(:,2),'-r')
58 ylabel('n_{BC|S} [kg/min]')
59 xlabel('Time [s]')
60
61 figure(2)
62 subplot(312)
63 hold on
64 %axis([0 40 -4*10^5 -3.5*10^5])
65 plot(t,x(:,1),'-b')
66 ylabel('N_BC [kg]')
67 xlabel('Time [s]')
68
69 figure(2)
70 subplot(313)
71 hold on
72 %axis([0 40 -4*10^5 -3.5*10^5])
73 plot(t,x(:,2),'-b')
74 ylabel('T_BC [C]')
75 xlabel('Time [s]')
76
77 figure(3)
78 subplot(311)
79 hold on
80 axis([0 1000 42 48])
81 plot(t,u(:,3),'-r')
82 ylabel('T_R [C]')
83 xlabel('Time [s]')
84
85 figure(3)
86 subplot(312)
87 hold on
88 axis([0 1000 2.8e4 2.9e4])
89 plot(t,x(:,1),'-b')
90 ylabel('N_BC [kg]')
91 xlabel('Time [s]')
92
93 figure(3)
94 subplot(313)
95 hold on
96 axis([0 1000 30 36])
97 plot(t,x(:,2),'-b')
```

```
ylabel('T_BC [C]')
xlabel('Time [s]')
```

A.1.2 Paramaters and constants

```
%% Process constants for RunCSTRmod.m

global c i

% Gamma - process "constants"
c.T_R =   43;     % C Reservoir NPK temperature
c.T_cin = 18;     % C Temperature coolingwater in
c.T_cout= 30;     % C Temperature coolingwater out
c.T_cmid = (c.T_cout-c.T_cin)/2; % C Mean cooling water temperature

% Initial values
i.N_BC  = 9.9*2.7*1.7*996.12*0.63; % kg (996.12 density NPK kg/m3)
i.T_BC  = 32;  % Celsius
i.n_R_BC= 210000/(60*60); % kg/min
i.n_BC_S= 210000/(60*60); % kg/min

% Steps on inputs
time_n_R_BC = 20;         % Steptime [min]
step_n_R_BC = 0;          % Stepfac from init value
time_n_BC_S = 20;         % Steptime [min]
step_n_BC_S = 0;          % Stepfac from init value
```

```
%% Process parameters for RunCSTRmod.m

global p c i

% Cp interpolation, from enthalpy Temperature diagram
% Linear approximation h = K_1 +aT

p.a_NPK1 = (32-9.5)/(42-32)*1000;

% Steady state calculation of U
p.A = (6.31-3.580+3.23-0.4)*1.515*128*2; % Total heat exchanger area. NB 3 not confirm
p.U = 1/(p.A*(i.T_BC-c.T_cmid))*(c.T_R-i.T_BC)*p.a_NPK1*i.n_R_BC; % KJ/m2Kh

p.Q =p.U*p.A*(i.T_BC-c.T_cmid)/4.184
```

A.1.3 S-function

```
function [sys,x0,str,ts] = sf_CSTRmod(t,x,u,flag)

```

```
global c p i
switch flag

    case 0  %Initialize
      sys = [2,      % number of continuous states
             0,      % number of discrete states
             2,      % number of outputs
             3,      % number of inputs
             0,      % reserved must be zero
             0,      % direct feedthrough flag
             1];     % number of sample times

        x0 = [i.N_BC i.T_BC]';
        str = [];
        ts  = [0 0];   % sample time: [period, offset]

    case 1 %Derivatives

        % State equations:
        dxdt(1) = u(1)-u(2);
        dxdt(2) = (1/x(1))*(u(3)-x(2))*u(1)...
            -(p.U*p.A/(x(1)*p.a_NPK1))*(x(2)-c.T_cmid);
        dxdt = dxdt(:);
        sys = dxdt;

    case 2 % Discrete state udate

        sys = []; %There is none

    case 3 % Outputs

        sys = [x(1) x(2)]';

    case 9 % Terminate

        sys = [];

    otherwise
        error(['unhandled flag = ',num2str(flag)]);

end
```

A.2 Model with lumped systems in series (CSTR in series)

The system i solved by use of Simulink, see figure 11

A.2.1 Run file

```
%% Published by Bjorn Tore Mathisen
% Model of the Bulk Cooler in NPK 4 as part of a specialization
```

```
% subject H2013.
% The following model approximates the Bulk Cooler as a CSTR:

clc
clf
clear all
close all

% Process constants:
constant

% Global parameters
parameters

%%
% Disturbances
% time_n_R_BC = 20;         % Steptime [min]
% step_n_R_BC = 0.01;          % Stepfac from init value
% time_n_BC_S = 20.5;        % Steptime [min]
% step_n_BC_S = 0.01;          % Stepfac from init value
time_T_R      = 20;
step_T_R      = 0.0;

%Starting simulink model
sim('CSTRseries.mdl');


disp('Total energy balance:')
text1 = ['Qwater: ',num2str(i.n_WR_W*4183*(x(end,1+i.n+3)-c.T_cin)),' J/s'];
disp(text1)
text2 = ['QNPK: ',num2str(i.n_BC_S*p.a_NPK1*(c.T_R-x(end,10+3))),' J/s'];
disp(text2)

figure(1)
subplot(211)
hold on
stages1 = [1:5];
axis([0 6 12 45])
title('Top section: Counter current Hx')
plot(stages1,x(end,stages1+2),'-xr')
plot([0 1],[c.T_R x(end,3)],'-xr')
plot(stages1,x(end,stages1+i.n+3),'-sb')
plot([5 6],[ x(end,5+i.n+3) x(end,10+i.n+3)],'-sb')
plot([0.5 0.5], [13 44],'--k')
plot([5.5 5.5], [13 44],'--k')
str(1) = {'T_{R}'};
text(0.1,38,str(1))
str(2) = {'Inlet'};
text(0.1,(45-12)/2+12,str(2))
str(6) = {'Inlet'};
text(5.6,(45-12)/2+12,str(6))
str(3) = {'Hx'};
text(0.6,(45-12)/2+12,str(3))
str(3) = {'Hx'};
```

```
57  text(5.1,(45-12)/2+12,str(3))
58  str(4) = {'T_{10}'};
59  text(5.6,16,str(4))
60  ylabel( 'T [C]')
61
62  figure(1)
63  subplot(212)
64  stages2 = [6:10];
65  hold on
66  axis([5 11 12 45])
67  title('Bottom section: Co-current Hx')
68  plot(stages2,x(end,stages2+3),'-xr')
69  plot( [5 6],[x(end,5+2) x(end,6+3)],'-xr')
70  plot(stages2,x(end,stages2+i.n+3),'-sb')
71  plot([5 6],[c.T_cin x(end,6+i.n+3)],'-sb')
72  plot([5.5 5.5], [13 44],'--k')
73  str(1) = {'T_{cin}'};
74  text(5.1,18,str(1))
75  str(2) = {'Inlet'};
76  text(5.1,(45-12)/2+12,str(2))
77  str(3) = {'Hx'};
78  text(5.6,(45-12)/2+12,str(3))
79  str(4) = {'T_{5}'};
80  text(5.1,38,str(4))
81  ylabel( 'T [C]')
82  xlabel('Stage')
83
84  % figure(2)
85  % subplot(211)
86  % stages2 = [1:12];
87  % hold on
88  % axis([1 12 15 45])
89  % plot(stages2,x(end,stages2+1),'sr')
90  % ylabel( 'T [C]')
91  % xlabel('Stage')
92  %
93  % figure(2)
94  % subplot(212)
95  % stages2 = [6:10];
96  % stages1 = [1:5]
97  % hold on
98  % axis([1 10 15 45])
99  % temp = [x(end,stages2+i.n+3) x(end,stages1+i.n+3)];
100 % plot([stages1 10 9 8 7 6 ],temp,'sb')
101 % ylabel( 'T [C]')
102 % xlabel('Stage')
```

A.2.2 Paramaters and constants

```
1  %% Process constants for RunCSTRmod.m
2
```

```
global c i

% Gamma - process "constants"
c.T_R =   43;     % C Reservoir NPK temperature
c.T_cin = 15;     % C Temperature coolingwater in
c.T_cout= 30;     % C Temperature coolingwater out
c.T_cmid = (c.T_cout-c.T_cin)/2; % C Mean cooling water temperature

%Number of interior stages
i.n = 10;     %Must be dividable by 2

% Solid section mass hold up calculation
i.N_i_s = ((6.31-3.580+3.23-0.4)/i.n)*1.834*1.516*950*0.63; %kg
i.N_i_l = 0.001*996.12;                          %kg
i.N_1_s = 1*1.834*1.516*950*0.63;                %kg
i.N_7_s = (3.580-3.230)*1.834*1.516*950*0.63;%kg

% Initial values
i.T_1  = 43;                % Celsius
i.n_R_BC= 210000/(60*60);   % kg/s
i.n_BC_S= 210000/(60*60);   % kg/s
i.n_WR_W= 50*996.12/(60*60);% kg/s ref 201 prod

% Steps on inputs
time_n_R_BC = 20;         % Steptime [min]
step_n_R_BC = 0;          % Stepfac from init value
time_n_BC_S = 20;         % Steptime [min]
step_n_BC_S = 0;          % Stepfac from init value
time_n_WR_W = 20;         % Steptime [min]
step_n_WR_W = 0;          % Stepfac from init value

% Testing:
% i.n_BC_S= 209.9*(1000/(60*60));   % kg/s
% i.n_WR_W = 58.6*(996.12/(60*60));
% c.T_R =   42.5;     % C Reservoir NPK temperature
% c.T_cin = 18.8;     % C Temperature coolingwater in
% c.T_cout= 32.3;     % C Temperature coolingwater out
% c.T_cmid = (c.T_cout-c.T_cin)/2; % C Mean cooling water temperature
```

```
%% Process parameters for RunCSTRmod.m

global p c i

% Cp interpolation, from enthalpy Temperature diagram
% Linear approximation h = K_1 +aT
p.beta = 0.6;
p.a_NPK1 = ((32-9.5)/(42-32)*1000)*p.beta;   %J/kgK
p.a_H2O1 = 4183;                      %J/kgK
p.alfa = 2.22;                          %Fitting parameter
p.gamma = 1;

% Steady state calculation of U
```

```
p.A = (6.31-3.580+3.23-0.4)*1.515*128*2*p.gamma; % Total heat exchanger area. m2
% NB 3 not confirmed height* width* plates
p.Ai = p.A/i.n; %Area per section m2
%p.Ui = (210000/60 *(0.38*4.1)*(2))/(p.Ai*(2))
p.U = ((i.n_WR_W*4183*15)/(p.A*((c.T_R-6)-c.T_cmid)))*p.alfa;
      % (m3/h*kg/m3/(min/h*s/min*J/kgK*K))/(m2*K) = J/sm2K

%Testing
 %p.a_NPK1 = 1880.8337;
```

A.2.3 S-function

```
function [sys,x0,str,ts] = sf_CSTRseries(t,x,u,flag)

global c p i
switch flag

    case 0  %Initialize
      sys = [3+2*i.n,      % number of continuous states
             0,       % number of discrete states
             3+2*i.n,       % number of outputs
             5,       % number of inputs
             0,       % reserved must be zero
             0,       % direct feedthrough flag
             1];      % number of sample times

        x0 = [i.N_1_s ones(1,i.n/2+1)*i.T_1 ones(1,i.n/2+1)*32 ...
            ones(1,i.n/2)*c.T_cin ones(1,i.n/2)*c.T_cin]';
        str = [];
        ts  = [0 0];   % sample time: [period, offset]

    case 1 %Derivatives

        % State equations:
             %Solids
        dxdt(1) = u(1)-u(2);  %Mass balance top section
        dxsdt(1) = (1/x(1))*(u(3)-x(2))*u(1); %Energy wrong top section
        s = [];  %Stage counter

        xl(1)= 0.0;         %Mock state for retaining index
        xl(i.n/2+2)=c.T_cin;%matching the solid phase
        for j = 2:1:i.n+3 %Extracting the state variables for solid EB
               xs(j-1) = x(j);
               if  j<i.n/2+3 & j>2
                    xl(j-1) = x(j+i.n+1);   %Counter current
               elseif j>i.n/2+3             %Skip middle section
                   xl(j-1) = x(j+i.n);      %Co-current
               end
        end
                    % EB Solid - Counter current Hx
                    % i.n = 10 -> Section 2-6
```

```
        for s = 2:1:i.n/2+1
              dxsdt(s) = (1/i.N_i_s)*(xs(s-1)-xs(s))*u(2)...
                     -(p.U*p.Ai/(i.N_i_s*p.a_NPK1))*(xs(s)-xl(s));

                     % Transfer back to program state vector
                     dxdt(s+1) = dxsdt(s);
        end

                     % EB Solid - Middle section
                     % i.n = 10 -> Section 7
        dxdt(i.n/2+3)= (1/i.N_7_s)*(xs(i.n/2+1)-xs(i.n/2+2))*u(2);

                     % EB Solid - Co-current Hx
                     % i.n = 10 -> Section 8-12
        s = [];
        for s = (i.n/2+3):1:i.n+2
             dxsdt(s) = (1/i.N_i_s)*(xs(s-1)-xs(s))*u(2)...
            -(p.U*p.Ai/(i.N_i_s*p.a_NPK1))*(xs(s)-xl(s));

            dxdt(s+1) = dxsdt(s);
        end
                     % EB Liquid - Co current HX
                     % i.n = 10 -> Section 8-12
        for s = (i.n/2+3):1:i.n+2
                dxldt(s) = (1/i.N_i_l)*(xl(s-1)-xl(s))*u(4)+...
                (p.U*p.Ai/(i.N_i_l*p.a_H2O1))*(xs(s)-xl(s));
                dxdt(s+1+i.n)=  dxldt(s);
        end
                     % EB Liquid - Counter current HX
                     % i.n = 10 -> Section 6
        s = i.n/2+1;
        dxldt(s) = (1/i.N_i_l)*(xl(i.n+2)-xl(s))*u(4)...
                 +(p.U*p.Ai/(i.N_i_l*p.a_H2O1))*(xs(s)-xl(s));
                    dxdt(s+2+i.n)=  dxldt(s);

                     % EB Liquid - Counter current HX
                     % i.n = 10 -> Section 2-5
        for s = i.n/2:-1:2
                dxldt(s) = (1/i.N_i_l)*(xl(s+1)-xl(s))*u(4)...
                 +(p.U*p.Ai/(i.N_i_l*p.a_H2O1))*(xs(s)-xl(s));
                dxdt(s+2+i.n)= dxldt(s);
        end
%                       s = 6
%                       disp('start')
%                       disp(dxldt(s))
%                       disp(xl(12))
%                       disp(xl(6))
%                       disp((1/(i.N_i_s)/5)*(xl(i.n+2)-xl(s))*u(4))
%                       disp((p.U*p.Ai/((i.N_i_s)/5*p.a_H2O1))*(xs(s)-xl(s)))
%


%         disp(dxldt)
%         disp(dxsdt)
```

```
%       disp('xl')
%      disp(xl')
%       disp(dxldt(6))
%          disp('xs')
%      disp(xs')
         dxdt = dxdt(:);
         sys = dxdt;

    case 2 % Discrete state udate

         sys = []; %There is none

    case 3 % Outputs

         j= [];
         for j=1:1:i.n*2+3
             sys(j) = [x(j)];
         end
         sys = sys';

    case 9 % Terminate

         sys = [];

    otherwise
         error(['unhandled flag = ',num2str(flag)]);

end
```

A.3 Modelling the phase changeover the span of $dT = 1°C$ (CSTR in series)

A.3.1 Run file

```
%% Published by Bjorn Tore Mathisen
% Model of the Bulk Cooler in NPK 4 as part of a specialization
% subject H2013.
% The following model approximates the Bulk Cooler as a CSTR:

clc
clf
clear all
close all

% Process constants:
constant

% Global parameters
parameters

%%
```

```
% Disturbances
% time_n_R_BC = 20;         % Steptime [min]
% step_n_R_BC = 0.01;          % Stepfac from init value
% time_n_BC_S = 20.5;         % Steptime [min]
% step_n_BC_S = 0.01;          % Stepfac from init value
time_T_R      = 20;
step_T_R      = 0.0;

%Starting simulink model
sim('CSTRseries.mdl');


disp('Total energy balance:')
text1 = ['Qwater: ',num2str(i.n_WR_W*4183*(x(end,1+i.n+3)-c.T_cin)),' J/s'];
disp(text1)
text2 = ['QNPK: ',num2str(i.n_BC_S*p.a_NPK1*(c.T_R-x(end,10+3))),' J/s'];
disp(text2)

figure(1)
subplot(211)
hold on
stages1 = [1:5];
axis([0 6 12 45])
title('Top section: Counter current Hx')
plot(stages1,x(end,stages1+2),'-xr')
plot([0 1],[c.T_R x(end,3)],'-xr')
plot(stages1,x(end,stages1+i.n+3),'-sb')
plot([5 6],[ x(end,5+i.n+3) x(end,10+i.n+3)],'-sb')
plot([0.5 0.5], [13 44],'--k')
plot([5.5 5.5], [13 44],'--k')
str(1) = {'T_{R}'};
text(0.1,38,str(1))
str(2) = {'Inlet'};
text(0.1,(45-12)/2+12,str(2))
str(6) = {'Inlet'};
text(5.6,(45-12)/2+12,str(6))
str(3) = {'Hx'};
text(0.6,(45-12)/2+12,str(3))
str(3) = {'Hx'};
text(5.1,(45-12)/2+12,str(3))
str(4) = {'T_{10}'};
text(5.6,16,str(4))
ylabel( 'T [C]')

figure(1)
subplot(212)
stages2 = [6:10];
hold on
axis([5 11 12 45])
title('Bottom section: Co-current Hx')
plot(stages2,x(end,stages2+3),'-xr')
plot( [5 6],[x(end,5+2) x(end,6+3)],'-xr')
plot(stages2,x(end,stages2+i.n+3),'-sb')
plot([5 6],[c.T_cin x(end,6+i.n+3)],'-sb')
```

```
plot([5.5 5.5], [13 44],'--k')
str(1) = {'T_{cin}'};
text(5.1,18,str(1))
str(2) = {'Inlet'};
text(5.1,(45-12)/2+12,str(2))
str(3) = {'Hx'};
text(5.6,(45-12)/2+12,str(3))
str(4) = {'T_{5}'};
text(5.1,38,str(4))
ylabel( 'T [C]')
xlabel('Stage')

% figure(2)
% subplot(211)
% stages2 = [1:12];
% hold on
% axis([1 12 15 45])
% plot(stages2,x(end,stages2+1),'sr')
% ylabel( 'T [C]')
% xlabel('Stage')
%
% figure(2)
% subplot(212)
% stages2 = [6:10];
% stages1 = [1:5]
% hold on
% axis([1 10 15 45])
% temp = [x(end,stages2+i.n+3) x(end,stages1+i.n+3)];
% plot([stages1 10 9 8 7 6 ],temp,'sb')
% ylabel( 'T [C]')
% xlabel('Stage')
```

A.3.2 Paramaters and constants

```
%% Process constants for RunCSTRmod.m

global c i

% Gamma - process "constants"
c.T_R =   43;    % C Reservoir NPK temperature
c.T_cin = 15;    % C Temperature coolingwater in
c.T_cout= 30;    % C Temperature coolingwater out
c.T_cmid = (c.T_cout-c.T_cin)/2; % C Mean cooling water temperature

%Number of interior stages
i.n = 10;    %Must be dividable by 2

% Solid section mass hold up calculation
i.N_i_s = ((6.31-3.580+3.23-0.4)/i.n)*1.834*1.516*950*0.63; %kg
i.N_i_l = 0.001*996.12;                        %kg
i.N_1_s = 1*1.834*1.516*950*0.63;              %kg
```

```
i.N_7_s = (3.580-3.230)*1.834*1.516*950*0.63;%kg

% Initial values
i.T_1  = 43;                % Celsius
i.n_R_BC= 210000/(60*60);   % kg/s
i.n_BC_S= 210000/(60*60);   % kg/s
i.n_WR_W= 50*996.12/(60*60);% kg/s ref 201 prod

% Steps on inputs
time_n_R_BC = 20;        % Steptime [min]
step_n_R_BC = 0;         % Stepfac from init value
time_n_BC_S = 20;        % Steptime [min]
step_n_BC_S = 0;         % Stepfac from init value
time_n_WR_W = 20;        % Steptime [min]
step_n_WR_W = 0;         % Stepfac from init value

% Testing:
% i.n_BC_S= 209.9*(1000/(60*60));   % kg/s
% i.n_WR_W = 58.6*(996.12/(60*60));
% c.T_R =   42.5;     % C Reservoir NPK temperature
% c.T_cin = 18.8;     % C Temperature coolingwater in
% c.T_cout= 32.3;     % C Temperature coolingwater out
% c.T_cmid = (c.T_cout-c.T_cin)/2; % C Mean cooling water temperature
```

```
%% Process parameters for RunCSTRmod.m

global p c i

% Cp interpolation, from enthalpy Temperature diagram
% Linear approximation h = K_1 +aT
p.beta = 0.68;
p.a_NPK1 = ((32-9.5)/(42-32)*1000)*p.beta;   %J/kgK
p.a_H2O1 = 4183;                    %J/kgK
p.alfa = 2.22;                        %Fitting parameter
p.gamma = 1;

% Steady state calculation of U
p.A = (6.31-3.580+3.23-0.4)*1.515*128*2*p.gamma; % Total heat exchanger area. m2
% NB 3 not confirmed height* width* plates
p.Ai = p.A/i.n; %Area per section m2
%p.Ui = (210000/60 *(0.38*4.1)*(2))/(p.Ai*(2))
p.U = ((i.n_WR_W*4183*15)/(p.A*((c.T_R-6)-c.T_cmid)))*p.alfa;
      % (m3/h*kg/m3/(min/h*s/min*J/kgK*K))/(m2*K) = J/sm2K

%Testing
x_temp = [25 32  34 34 40 42 54];        % Temperature
y_enth = [0 9.5 13 23 30 33 50];        % Enthalpy
p.a_NPKbot = ((13-0)/(33-25)*1000*p.beta);
p.a_NPKphase = ((23-13)/(34-33)*1000*p.beta);
p.a_NPKtop = ((50-23)/(54-34)*1000*p.beta);
```

A.3.3 S-function

```
function [sys,x0,str,ts] = sf_CSTRseries(t,x,u,flag)

global c p i
switch flag

    case 0  %Initialize
      sys = [3+2*i.n,      % number of continuous states
            0,       % number of discrete states
            3+2*i.n,       % number of outputs
            5,       % number of inputs
            0,       % reserved must be zero
            0,       % direct feedthrough flag
            1];      % number of sample times

        x0 = [i.N_1_s ones(1,i.n/2+1)*i.T_1 ones(1,i.n/2+1)*32 ...
            ones(1,i.n/2)*c.T_cin ones(1,i.n/2)*c.T_cin]';
        str = [];
        ts  = [0 0];   % sample time: [period, offset]

    case 1 %Derivatives

        % State equations:
             %Solids
        dxdt(1) = u(1)-u(2);  %Mass balance top section
        dxsdt(1) = (1/x(1))*(u(3)-x(2))*u(1); %Energy wrong top section
        s = [];  %Stage counter

        xl(1)= 0.0;          %Mock state for retaining index
        xl(i.n/2+2)=c.T_cin;%matching the solid phase
        for j = 2:1:i.n+3 %Extracting the state variables for solid EB
               xs(j-1) = x(j);
               if  j<i.n/2+3 & j>2
                    xl(j-1) = x(j+i.n+1);    %Counter current
               elseif j>i.n/2+3              %Skip middle section
                   xl(j-1) = x(j+i.n);       %Co-current
               end
        end
                    % EB Solid - Counter current Hx
                    % i.n = 10 -> Section 2-6
        for s = 2:1:i.n/2+1
            if x(s) > 34
                p.a_NPK1 =p.a_NPKtop;
            elseif x(s) > 33
                 p.a_NPK1 =p.a_NPKphase;
            else
                p.a_NPK1 =p.a_NPKbot;
            end
              dxsdt(s) = (1/i.N_i_s)*(xs(s-1)-xs(s))*u(2)...
                    -(p.U*p.Ai/(i.N_i_s*p.a_NPK1))*(xs(s)-xl(s));

```

```
51                      % Transfer back to program state vector
52                      dxdt(s+1) = dxsdt(s);
53          end
54
55                      % EB Solid - Middle section
56                      % i.n = 10 -> Section 7
57          dxdt(i.n/2+3)= (1/i.N_7_s)*(xs(i.n/2+1)-xs(i.n/2+2))*u(2);
58
59                      % EB Solid - Co-current Hx
60                      % i.n = 10 -> Section 8-12
61          s = [];
62          for s = (i.n/2+3):1:i.n+2
63           if x(s) > 34
64                  p.a_NPK1 =p.a_NPKtop;
65              elseif x(s) > 33
66                   p.a_NPK1 =p.a_NPKphase;
67              else
68                  p.a_NPK1 =p.a_NPKbot;
69              end
70               dxsdt(s) = (1/i.N_i_s)*(xs(s-1)-xs(s))*u(2)...
71              -(p.U*p.Ai/(i.N_i_s*p.a_NPK1))*(xs(s)-xl(s));
72
73              dxdt(s+1) = dxsdt(s);
74          end
75                      % EB Liquid - Co current HX
76                      % i.n = 10 -> Section 8-12
77          for s = (i.n/2+3):1:i.n+2
78                  dxldt(s) = (1/i.N_i_l)*(xl(s-1)-xl(s))*u(4)+...
79                  (p.U*p.Ai/(i.N_i_l*p.a_H2O1))*(xs(s)-xl(s));
80                  dxdt(s+1+i.n)=  dxldt(s);
81          end
82                      % EB Liquid - Counter current HX
83                      % i.n = 10 -> Section 6
84          s = i.n/2+1;
85          dxldt(s) = (1/i.N_i_l)*(xl(i.n+2)-xl(s))*u(4)...
86                   +(p.U*p.Ai/(i.N_i_l*p.a_H2O1))*(xs(s)-xl(s));
87                      dxdt(s+2+i.n)=  dxldt(s);
88
89                      % EB Liquid - Counter current HX
90                      % i.n = 10 -> Section 2-5
91          for s = i.n/2:-1:2
92                  dxldt(s) = (1/i.N_i_l)*(xl(s+1)-xl(s))*u(4)...
93                   +(p.U*p.Ai/(i.N_i_l*p.a_H2O1))*(xs(s)-xl(s));
94                  dxdt(s+2+i.n)= dxldt(s);
95          end
96  %                       s = 6
97  %                       disp('start')
98  %                       disp(dxldt(s))
99  %                       disp(xl(12))
100 %                       disp(xl(6))
101 %                       disp((1/(i.N_i_s)/5)*(xl(i.n+2)-xl(s))*u(4))
102 %                       disp((p.U*p.Ai/((i.N_i_s)/5*p.a_H2O1))*(xs(s)-xl(s)))
103 %
104
```

```
105
106 %          disp(dxldt)
107 %          disp(dxsdt)
108 %       disp('xl')
109 %      disp(xl')
110 %       disp(dxldt(6))
111 %          disp('xs')
112 %      disp(xs')
113           dxdt = dxdt(:);
114           sys = dxdt;
115
116       case 2 % Discrete state udate
117
118           sys = []; %There is none
119
120       case 3 % Outputs
121
122           j= [];
123           for j=1:1:i.n*2+3
124               sys(j) = [x(j)];
125           end
126           sys = sys';
127
128       case 9 % Terminate
129
130           sys = [];
131
132       otherwise
133           error(['unhandled flag = ',num2str(flag)]);
134
135 end
```

A.4 Modelling a single counter current heat exchanger (CSTR in series)

A.4.1 Run file

```
1  %% Published by Bjorn Tore Mathisen
2  % Model of the Bulk Cooler in NPK 4 as part of a specialization
3  % subject H2013.
4  % The following model approximates the Bulk Cooler as a CSTR:
5
6  clc
7  clf
8  clear all
9  close all
10 hold on
11 % Process constants:
12 constant
13
14 % Global parameters
```

```
parameters

%%
% Disturbances
% time_n_R_BC = 20;         % Steptime [min]
% step_n_R_BC = 0.01;          % Stepfac from init value
% time_n_BC_S = 20.5;         % Steptime [min]
% step_n_BC_S = 0.01;          % Stepfac from init value
time_T_R      = 20;
step_T_R      = 0.0;

%Starting simulink model
sim('CSTRseriessplitcc.mdl');

disp('Total energy balance:')
text1 = ['Qwater: ',num2str(i.n_WR_W*4183*(xlmot(end,1)-c.T_cin)),' J/s'];
disp(text1)
text2 = ['QNPK:   ',num2str(i.n_BC_S*p.a_NPK1*(c.T_R-xsmot(end,12))),' J/s'];
disp(text2)


figure(1)
hold on
stages1=[1:10];
axis([0 11 12 45])
plot([0:10],[c.T_R xsmot(end,stages1+2)],'-xr')
plot([1:11],[xlmot(end,stages1) c.T_cin],'-sb')
ylabel( 'T [C]')
xlabel('Stage')

plot([0.5 0.5], [13 44],'--k')
plot([10.5 10.5], [13 45],'--k')
str(1) = {'T_{R}'};
text(0.1,38,str(1))
str(2) = {'In'};
text(0.1,(45-12)/2+12,str(2))
str(6) = {'In'};
text(10.6,(45-12)/2+12,str(6))
str(3) = {'Hx'};
text(0.6,(45-12)/2+12,str(3))
str(3) = {'Hx'};
text(10.0,(45-12)/2+12,str(3))
str(4) = {'T_{10}'};
text(10.6,20,str(4))
ylabel( 'T [C]')
% figure(2)
% subplot(211)
% stages2 = [1:12];
% hold on
% axis([1 12 15 45])
% plot(stages2,x(end,stages2+1),'sr')
% ylabel( 'T [C]')
% xlabel('Stage')
%
```

```
% figure(2)
% subplot(212)
% stages2 = [6:10];
% stages1 = [1:5]
% hold on
% axis([1 10 15 45])
% temp = [x(end,stages2+i.n+3) x(end,stages1+i.n+3)]
% plot([stages1 10 9 8 7 6 ],temp,'sb')
% ylabel( 'T [C]')
% xlabel('Stage')
%
% figure(1)
% subplot(312)
% hold on
% axis([0 40 -4*10^5 -3.5*10^5])
% plot(t,x(:,1),'-b')
% ylabel('N_{BC} [kg]')
% xlabel('Time [min]')
%
% figure(1)
% subplot(313)
% hold on
% axis([0 40 -4*10^5 -3.5*10^5])
% plot(t,x(:,2),'-b')
% ylabel('T_{BC} [C]')
% xlabel('Time [min]')
%
% figure(2)
% subplot(311)
% hold on
% axis([0 40 -4*10^5 -3.5*10^5])
% plot(t,u(:,2),'-r')
% ylabel('n_{BC|S} [kg/min]')
% xlabel('Time [min]')
%
% figure(2)
% subplot(312)
% hold on
% axis([0 40 -4*10^5 -3.5*10^5])
% plot(t,x(:,1),'-b')
% ylabel('N_BC [kg]')
% xlabel('Time [min]')
%
% figure(2)
% subplot(313)
% hold on
% axis([0 40 -4*10^5 -3.5*10^5])
% plot(t,x(:,2),'-b')
% ylabel('T_BC [C]')
% xlabel('Time [min]')
%
% figure(3)
% subplot(311)
% hold on
```

```
123 % axis([0 40 -4*10^5 -3.5*10^5])
124 % plot(t,u(:,3),'-r')
125 % ylabel('T_BC [C]')
126 % xlabel('Time [min]')
127 %
128 % figure(3)
129 % subplot(312)
130 % hold on
131 % axis([0 40 -4*10^5 -3.5*10^5])
132 % plot(t,x(:,1),'-b')
133 % ylabel('N_BC [kg]')
134 % xlabel('Time [min]')
135 %
136 % figure(3)
137 % subplot(313)
138 % hold on
139 % axis([0 40 -4*10^5 -3.5*10^5])
140 % plot(t,x(:,2),'-b')
141 % ylabel('T_BC [C]')
142 % xlabel('Time [min]')
```

A.4.2 Paramaters and constants

```
1  %% Process constants for RunCSTRmod.m
2
3  global c i
4
5  % Gamma - process "constants"
6  c.T_R =   43;    % C Reservoir NPK temperature
7  c.T_cin = 15;    % C Temperature coolingwater in
8  c.T_cout= 30;    % C Temperature coolingwater out
9  c.T_cmid = (c.T_cout-c.T_cin)/2; % C Mean cooling water temperature
10
11 %Number of interior stages
12 i.n = 10;    %Must be dividable by 2
13
14 % Solid section mass hold up calculation
15 i.N_i_s = ((6.31-3.580+3.23-0.4)/i.n)*1.834*1.516*950*0.63; %kg
16 i.N_i_l = 0.001*996.12;                        %kg
17 i.N_1_s = 1*1.834*1.516*950*0.63;              %kg
18 i.N_7_s = (3.580-3.230)*1.834*1.516*950*0.63;%kg
19
20 % Initial values
21 i.T_1  = 43;                % Celsius
22 i.n_R_BC= 210000/(60*60);   % kg/s
23 i.n_BC_S= 210000/(60*60);   % kg/s
24 i.n_WR_W= 50*996.12/(60*60);% kg/s ref 201 prod
25
26 % Steps on inputs
27 time_n_R_BC = 20;        % Steptime [min]
28 step_n_R_BC = 0;         % Stepfac from init value
```

```
time_n_BC_S = 20;         % Steptime [min]
step_n_BC_S = 0;          % Stepfac from init value
time_n_WR_W = 20;         % Steptime [min]
step_n_WR_W = 0;          % Stepfac from init value

% Testing:
% i.n_BC_S= 209.9*(1000/(60*60));   % kg/s
% i.n_WR_W = 58.6*(996.12/(60*60));
% c.T_R =   42.5;     % C Reservoir NPK temperature
% c.T_cin = 18.8;     % C Temperature coolingwater in
% c.T_cout= 32.3;     % C Temperature coolingwater out
% c.T_cmid = (c.T_cout-c.T_cin)/2; % C Mean cooling water temperature
```

```
%% Process parameters for RunCSTRmod.m

global p c i

% Cp interpolation, from enthalpy Temperature diagram
% Linear approximation h = K_1 +aT
p.beta = 0.6;
p.a_NPK1 = ((32-9.5)/(42-32)*1000)*p.beta;   %J/kgK
p.a_H2O1 = 4183;                     %J/kgK
p.alfa = 2.22;                        %Fitting parameter
p.gamma = 1;

% Steady state calculation of U
p.A = (6.31-3.580+3.23-0.4)*1.515*128*2*p.gamma; % Total heat exchanger area. m2
% NB 3 not confirmed height* width* plates
p.Ai = p.A/i.n; %Area per section m2
%p.Ui = (210000/60 *(0.38*4.1)*(2))/(p.Ai*(2))
p.U = ((i.n_WR_W*4183*15)/(p.A*((c.T_R-6)-c.T_cmid)))*p.alfa;
      % (m3/h*kg/m3/(min/h*s/min*J/kgK*K))/(m2*K) = J/sm2K

%Testing
 %p.a_NPK1 = 1880.8337;
```

A.4.3 S-function

```
function [sys,x0,str,ts] = sf_CSTRseries(t,x,u,flag)

global c p i
switch flag

    case 0  %Initialize
      sys = [2+i.n*2,      % number of continuous states
             0,       % number of discrete states
             2+i.n*2,       % number of outputs
             5,       % number of inputs
             0,       % reserved must be zero
```

```
12              0,       % direct feedthrough flag
13              1];      % number of sample times
14
15          x0 = [i.N_1_s 43 42 41 40 38.9 37.8 36.7 35.5 34.3 33.1 31.9
    ...
16              ones(1,i.n)*c.T_cin]';
17          str = [];
18          ts  = [0 0];   % sample time: [period, offset]
19
20      case 1 %Derivatives
21
22          % State equations:
23               %Solids
24          dxdt(1) = u(1)-u(2);  %Mass balance top section
25          dxsdt(1) = (1/x(1))*(u(3)-x(2))*u(1); %Energy wrong top section
26          s = [];  %Stage counter
27
28          xl(1)= 0.0;         %Mock state for retaining index
29          %xl(i.n/2+2)=c.T_cin;%matching the solid phase
30          for j = 2:1:i.n+2 %Extracting the state variables for solid EB
31                 xs(j-1) = x(j);
32                 if  j>2
33                      xl(j-1) = x(j+i.n);   %Counter current
34                 end
35          end
36                      % EB Solid - Counter current Hx
37                      % i.n = 10 -> Section 2-11
38          for s = 2:1:i.n+1
39                dxsdt(s) = (1/i.N_i_s)*(xs(s-1)-xs(s))*u(2)...
40                       -(p.U*p.Ai/(i.N_i_s*p.a_NPK1))*(xs(s)-xl(s));
41
42                      % Transfer back to program state vector
43                      dxdt(s+1) = dxsdt(s);
44          end
45                      % EB Liquid - Counter current HX
46                      % i.n = 10 -> Section 11
47
48          s = i.n+1;
49                  dxldt(s) = (1/i.N_i_l)*(u(5)-xl(s))*u(4)...
50                   +(p.U*p.Ai/(i.N_i_l*p.a_H2O1))*(xs(s)-xl(s));
51                  dxdt(i.n+1+s)= dxldt(s);
52
53                      % EB Liquid - Counter current HX
54                      % i.n = 10 -> Section 2-10
55          for s = i.n:-1:2
56                  dxldt(s) = (1/i.N_i_l)*(xl(s+1)-xl(s))*u(4)...
57                   +(p.U*p.Ai/(i.N_i_l*p.a_H2O1))*(xs(s)-xl(s));
58                  dxdt(i.n+1+s)= dxldt(s);
59          end
60
61          dxdt = dxdt(:);
62          sys = dxdt;
63
64      case 2 % Discrete state udate
```

```
65
66          sys = []; %There is none
67
68      case 3 % Outputs
69
70          j= [];
71          for j=1:1:i.n*2+2
72              sys(j) = [x(j)];
73          end
74          sys = sys';
75
76      case 9 % Terminate
77
78          sys = [];
79
80      otherwise
81          error(['unhandled flag = ',num2str(flag)]);
82
83  end
```

With phase change:

```
1   function [sys,x0,str,ts] = sf_CSTRseries(t,x,u,flag)
2
3   global c p i
4   switch flag
5
6       case 0  %Initialize
7         sys = [2+i.n*2,      % number of continuous states
8                0,       % number of discrete states
9                2+i.n*2,       % number of outputs
10               5,       % number of inputs
11               0,       % reserved must be zero
12               0,       % direct feedthrough flag
13               1];      % number of sample times
14
15          x0 = [i.N_1_s 43 42 41 40 38.9 37.8 36.7 35.5 34.3 33.1 31.9
    |...                                                                  |
16              ones(1,i.n)*c.T_cin]';
17          str = [];
18          ts  = [0 0];   % sample time: [period, offset]
19
20      case 1 %Derivatives
21
22          % State equations:
23               %Solids
24          dxdt(1) = u(1)-u(2);  %Mass balance top section
25          dxsdt(1) = (1/x(1))*(u(3)-x(2))*u(1); %Energy wrong top section
26          s = [];  %Stage counter
27
28          xl(1)= 0.0;          %Mock state for retaining index
29          %xl(i.n/2+2)=c.T_cin;%matching the solid phase
30          for j = 2:1:i.n+2 %Extracting the state variables for solid EB
```

```
31                  xs(j-1) = x(j);
32                  if  j>2
33                       xl(j-1) = x(j+i.n);   %Counter current
34                  end
35          end
36                       % EB Solid - Counter current Hx
37                       % i.n = 10 -> Section 2-11
38          for s = 2:1:i.n+1
39                 if x(s) > 34
40                  p.a_NPK1 =p.a_NPKtop;
41              elseif x(s) > 33
42                   p.a_NPK1 =p.a_NPKphase;
43              else
44                  p.a_NPK1 =p.a_NPKbot;
45              end
46                dxsdt(s) = (1/i.N_i_s)*(xs(s-1)-xs(s))*u(2)...
47                       -(p.U*p.Ai/(i.N_i_s*p.a_NPK1))*(xs(s)-xl(s));
48
49                       % Transfer back to program state vector
50                       dxdt(s+1) = dxsdt(s);
51          end
52                       % EB Liquid - Counter current HX
53                       % i.n = 10 -> Section 11
54
55          s = i.n+1;
56                  dxldt(s) = (1/i.N_i_l)*(u(5)-xl(s))*u(4)...
57                   +(p.U*p.Ai/(i.N_i_l*p.a_H2O1))*(xs(s)-xl(s));
58                  dxdt(i.n+1+s)= dxldt(s);
59
60                       % EB Liquid - Counter current HX
61                       % i.n = 10 -> Section 2-10
62          for s = i.n:-1:2
63                  dxldt(s) = (1/i.N_i_l)*(xl(s+1)-xl(s))*u(4)...
64                   +(p.U*p.Ai/(i.N_i_l*p.a_H2O1))*(xs(s)-xl(s));
65                  dxdt(i.n+1+s)= dxldt(s);
66          end
67
68          dxdt = dxdt(:);
69          sys = dxdt;
70
71      case 2 % Discrete state udate
72
73          sys = []; %There is none
74
75      case 3 % Outputs
76
77          j= [];
78          for j=1:1:i.n*2+2
79              sys(j) = [x(j)];
80          end
81          sys = sys';
82
83      case 9 % Terminate
84
```

```
85          sys = [];
86
87      otherwise
88          error(['unhandled flag = ',num2str(flag)]);
89
90  end
```

A.5 Dynamic model with PI control

The system i solved by use of Simulink, see figure 19

A.5.1 Run file

```
1  %% Published by Bjorn Tore Mathisen
2  % Model of the Bulk Cooler in NPK 4 as part of a specialization
3  % subject H2013.
4  % The following model approximates the Bulk Cooler as a CSTR:
5
6  clc
7  clf
8  clear all
9  close all
10
11 % Process constants:
12 constant
13
14 % Global parameters
15 parameters
16
17 %%
18 % Disturbances
19  time_n_R_BC = 1000;         % Steptime [min]
20  step_n_R_BC = 40000/(60*60);;          % Stepfac from init value
21  time_n_BC_S = 1000;         % Steptime [min]
22  step_n_BC_S = 40000/(60*60);;          % Stepfac from init value
23 time_T_R       = 1000;
24 step_T_R       = 4.6;
25 time_T_cin       = 20;
26 step_T_cin     = 0.0;
27 time_n_WR_W    = 20;
28 step_n_WR_W    = 0.0;
29
30 %Starting simulink model
31 sim('CSTRseries.mdl');
32
33
34 disp('Total energy balance:')
35 text1 = ['Qwater: ',num2str(i.n_WR_W*4183*(x(end,1+i.n+3)-c.T_cin)),' J/s'];
36 disp(text1)
37 text2 = ['QNPK: ',num2str(i.n_BC_S*p.a_NPK1*(c.T_R-x(end,10+3))),' J/s'];
```

```
disp(text2)

figure(1)
subplot(211)
hold on
stages1 = [1:5];
axis([0 6 12 48])
title('Top section: Counter current Hx')
plot(stages1,x(end,stages1+2),'-xr')
plot([0 1],[x(end,2) x(end,3)],'-xr')
plot(stages1,x(end,stages1+i.n+3),'-sb')
plot([5 6],[ x(end,5+i.n+3) x(end,10+i.n+3)],'-sb')
plot([0.5 0.5], [13 44],'--k')
plot([5.5 5.5], [13 44],'--k')
str(1) = {'T_{R}'};
text(0.1,38,str(1))
str(2) = {'Inlet'};
text(0.1,(45-12)/2+12,str(2))
str(6) = {'Inlet'};
text(5.6,(45-12)/2+12,str(6))
str(3) = {'Hx'};
text(0.6,(45-12)/2+12,str(3))
str(3) = {'Hx'};
text(5.1,(45-12)/2+12,str(3))
str(4) = {'T_{10}'};
text(5.6,16,str(4))
ylabel( 'T [C]')

figure(1)
subplot(212)
stages2 = [6:10];
hold on
axis([5 11 12 43])
title('Bottom section: Co-current Hx')
plot(stages2,x(end,stages2+3),'-xr')
plot( [5 6],[x(end,5+2) x(end,6+3)],'-xr')
plot(stages2,x(end,stages2+i.n+3),'-sb')
plot([5 6],[c.T_cin x(end,6+i.n+3)],'-sb')
plot([5.5 5.5], [13 44],'--k')
str(1) = {'T_{cin}'};
text(5.1,18,str(1))
str(2) = {'Inlet'};
text(5.1,(45-12)/2+12,str(2))
str(3) = {'Hx'};
text(5.6,(45-12)/2+12,str(3))
str(4) = {'T_{5}'};
text(5.1,38,str(4))
ylabel( 'T [C]')
xlabel('Stage')

figure(3)
subplot(211)
hold on
axis([0 1000 13.5 16])
```

```
plot(t,u(:,4),'-r')
plot([600 600], [i.n_WR_W i.n_WR_W+i.n_WR_W*0.1],'-k')
str(1) = {'\Delta u(\infty)'};
text(610,14.5,str(1))
ylabel('n_{WR|BC} [kg/s]')
xlabel('Time [s]')

figure(3)
subplot(212)
hold on
axis([0 1000 29.4 32.4])
plot(t,x(:,13),'-b')

plot([20 106.8], [x(end,13) x(end,13)],'-k')
str(1) = {'\theta'};
text(63,x(end,13)-0.2,str(1))

T63 = (x(end,13)-x(1,13))*0.63+x(1,13); %32.6741
plot([0 240.2], [T63 T63],':k')
plot([240.2 240.2], [T63 x(1,13)],':k')
str(2) = {'63%'};
text(120,T63-0.3,str(2))

plot([106.8 240.2], [x(end,13) x(end,13)],'-k')
plot([106.8  106.8 ], [x(end,13)-0.1 x(end,13)+0.1],'-k')
str(1) = {'\tau_1'};
text(175,x(end,13)-0.2,str(1))

plot([600 600], [x(end,13) x(1,13)],'-k')
str(1) = {'\Delta y(\infty)'};
text(610,31,str(1))

str(1) = {'\Delta y(t)'};
text(160,31.5,str(1))

ylabel('T_{BC|S} [C]')
xlabel('Time [s]')

tau_1 = 240.2-106.8
tc = 8*(-20+106.8)
Dy = x(end,13)-x(1,13)
Du = u(end,4)-u(1,4)
k = (x(end,13)-x(1,13))/(u(end,4)-u(1,4))
K_C = 1/k*(tau_1/(2*80))

figure(2)
subplot(311)
hold on
%axis([0 1000 42.7 43.7])
plot(t,u(:,3),'-g')
ylabel('T_{R} [C]')
xlabel('Time [s]')

figure(2)
```

```
146 subplot(312)
147 hold on
148 %axis([0 1000 13 18])
149 plot(t,u(:,4),'-r')
150 ylabel('n_{WR|BC} [kg/s]')
151 xlabel('Time [s]')
152
153 figure(2)
154 subplot(313)
155 hold on
156 %axis([0 1000 31.5 33.5])
157 plot(t,x(:,13),'-b')
158 ylabel('T_{BC|S} [C]')
159 xlabel('Time [s]')
160
161 figure(4)
162 subplot(211)
163 hold on
164 axis([0 1000 13.5 16])
165 plot(t,u(:,4),'-r')
166 plot([600 600], [i.n_WR_W i.n_WR_W+i.n_WR_W*0.1],'-k')
167 str(1) = {'\Delta u(\infty)'};
168 text(610,14.5,str(1))
169 ylabel('n_{WR|BC} [kg/s]')
170 xlabel('Time [s]')
171
172 figure(4)
173 subplot(212)
174 hold on
175 axis([0 1000 29.4 32.4])
176 plot(t,x(:,13),'-b')
177
178 plot([20 106.8], [x(end,13) x(end,13)],'-k')
179 str(1) = {'\theta'};
180 text(63,x(end,13)-0.2,str(1))
181
182 T63 = (30.8-x(1,13))*0.63+x(1,13); %32.6741
183 plot([0 136.8], [T63 T63],':k')
184 plot([136.8 136.8], [T63 x(1,13)],':k')
185 str(2) = {'63%'};
186 text(68,T63-0.3,str(2))
187
188 plot([106.8 136.8], [x(end,13) x(end,13)],'-k')
189 plot([106.8  106.8 ], [x(end,13)-0.1 x(end,13)+0.1],'-k')
190 str(1) = {'\tau_1'};
191 text(115,x(end,13)-0.2,str(1))
192
193 plot([212 212], [30.8 x(1,13)],'-k')
194 str(1) = {'\Delta y(\infty)'};
195 text(215,31.5,str(1))
196
197 str(1) = {'\Delta y(t)'};
198 text(40,31.5,str(1))
199
```

```
ylabel('T_{BC|S} [C]')
xlabel('Time [s]')

tau_1 = 136.8-106.8
tc = 8*(-20+106.8)
Dy = 30.8-x(1,13)
Du = u(end,4)-u(1,4)
k = (30.8-x(1,13))/(u(end,4)-u(1,4))
K_C = 1/k*(tau_1/(2*80))
```

A.5.2 Paramaters and constants

```
%% Process constants for RunCSTRmod.m

global c i

% Gamma - process "constants"
c.T_R =   43;     % C Reservoir NPK temperature
c.T_cin = 15;     % C Temperature coolingwater in
c.T_cout= 30;     % C Temperature coolingwater out
c.T_cmid = (c.T_cout-c.T_cin)/2; % C Mean cooling water temperature

%Number of interior stages
i.n = 10;     %Must be dividable by 2

% Solid section mass hold up calculation
i.N_i_s = ((6.31-3.580+3.23-0.4)/i.n)*1.834*1.516*950*0.63; %kg
i.N_i_l = 0.001*996.12;                          %kg
i.N_1_s = 1*1.834*1.516*950*0.63;                %kg
i.N_7_s = (3.580-3.230)*1.834*1.516*950*0.63;%kg

% Initial values
i.T_1  = 43;                % Celsius
i.n_R_BC= 210000/(60*60);   % kg/s
i.n_BC_S= 210000/(60*60);   % kg/s
i.n_WR_W= 50*996.12/(60*60);% kg/s ref 201 prod
%i.n_WR_W = 15.7212

% Steps on inputs
time_n_R_BC = 20;        % Steptime [min]
step_n_R_BC = 0;         % Stepfac from init value
time_n_BC_S = 20;        % Steptime [min]
step_n_BC_S = 0;         % Stepfac from init value
time_n_WR_W = 20;        % Steptime [min]
step_n_WR_W = 0;         % Stepfac from init value
```

```
%% Process parameters for RunCSTRmod.m

global p c i
```

```
4
5  % Cp interpolation, from enthalpy Temperature diagram
6  % Linear approximation h = K_1 +aT
7  p.beta = 0.68;
8  p.a_NPK1 = ((32-9.5)/(42-32)*1000)*p.beta;   %J/kgK
9  p.a_H2O1 = 4183;                    %J/kgK
10 p.alfa = 2.22;                       %Fitting parameter
11 p.gamma = 1;
12
13 % Steady state calculation of U
14 p.A = (6.31-3.580+3.23-0.4)*1.515*128*2*p.gamma; % Total heat exchanger area. m2
15 % NB 3 not confirmed height* width* plates
16 p.Ai = p.A/i.n; %Area per section m2
17 %p.Ui = (210000/60 *(0.38*4.1)*(2))/(p.Ai*(2))
18 p.U = ((i.n_WR_W*4183*15)/(p.A*((c.T_R-6)-c.T_cmid)))*p.alfa;
19       % (m3/h*kg/m3/(min/h*s/min*J/kgK*K))/(m2*K) = J/sm2K
20
21 %Testing
22 x_temp = [25 32  34 34 40 42 54];        % Temperature
23 y_enth = [0 9.5 13 23 30 33 50];        % Enthalpy
24 p.a_NPKbot = ((13-0)/(33-25)*1000*p.beta);
25 p.a_NPKphase = ((23-13)/(34-33)*1000*p.beta);
26 p.a_NPKtop = ((50-23)/(54-34)*1000*p.beta);
```

A.5.3 S-function

```
1  function [sys,x0,str,ts] = sf_CSTRseries(t,x,u,flag)
2
3  global c p i
4  switch flag
5
6      case 0  %Initialize
7        sys = [3+2*i.n,      % number of continuous states
8               0,       % number of discrete states
9               3+2*i.n,       % number of outputs
10              5,       % number of inputs
11              0,       % reserved must be zero
12              0,       % direct feedthrough flag
13              1];      % number of sample times
14
15         x0 = [ i.N_i_s 43 41.5 40.02 38.55 37.09 35.64...
16             35.64 33.6 33.36 33.14 32.95 31.98...
17             29.22 27.85 26.48 25.12 23.76...
18             16.89 18.56 20.04 21.35 22.42]';
19         str = [];
20         ts  = [0 0];   % sample time: [period, offset]
21
22     case 1 %Derivatives
23         % State equations:
24              %Solids
25         dxdt(1) = u(1)-u(2);  %Mass balance top section
```

```
26          dxsdt(1) = (1/x(1))*(u(3)-x(2))*u(1); %Energy top section
27          dxdt(2) = dxsdt(1);
28          s = [];  %Stage counter
29
30          xl(1)= 0.0;          %Mock state for retaining index
31          xl(i.n/2+2)=u(5);%matching the solid phase
32          for j = 2:1:i.n+3 %Extracting the state variables for solid EB
33                 xs(j-1) = x(j);
34                 if  j<i.n/2+3 & j>2
35                      xl(j-1) = x(j+i.n+1);    %Counter current
36                 elseif j>i.n/2+3              %Skip middle section
37                     xl(j-1) = x(j+i.n);       %Co-current
38                 end
39          end
40                      % EB Solid - Counter current Hx
41                      % i.n = 10 -> Section 2-6
42          for s = 2:1:i.n/2+1
43              if x(s) > 34
44                  p.a_NPK1 =p.a_NPKtop;
45              elseif x(s) > 33
46                   p.a_NPK1 =p.a_NPKphase;
47              else
48                  p.a_NPK1 =p.a_NPKbot;
49              end
50                dxsdt(s) = (1/i.N_i_s)*(xs(s-1)-xs(s))*u(2)...
51                       -(p.U*p.Ai/(i.N_i_s*p.a_NPK1))*(xs(s)-xl(s));
52
53                      % Transfer back to program state vector
54                      dxdt(s+1) = dxsdt(s);
55          end
56
57                      % EB Solid - Middle section
58                      % i.n = 10 -> Section 7
59          dxdt(i.n/2+3)= (1/i.N_7_s)*(xs(i.n/2+1)-xs(i.n/2+2))*u(2);
60
61                      % EB Solid - Co-current Hx
62                      % i.n = 10 -> Section 8-12
63          s = [];
64
65          for s = (i.n/2+3):1:i.n+2
66         if x(s) > 34
67                  p.a_NPK1 =p.a_NPKtop;
68              elseif x(s) > 33
69                   p.a_NPK1 =p.a_NPKphase;
70              else
71                  p.a_NPK1 =p.a_NPKbot;
72          end
73               dxsdt(s) = (1/i.N_i_s)*(xs(s-1)-xs(s))*u(2)...
74              -(p.U*p.Ai/(i.N_i_s*p.a_NPK1))*(xs(s)-xl(s));
75
76              dxdt(s+1) = dxsdt(s);
77          end
78                      % EB Liquid - Co current HX
79                      % i.n = 10 -> Section 8-12
```

```
80          for s = (i.n/2+3):1:i.n+2
81                  dxldt(s) = (1/i.N_i_l)*(xl(s-1)-xl(s))*u(4)+...
82                  (p.U*p.Ai/(i.N_i_l*p.a_H2O1))*(xs(s)-xl(s));
83                  dxdt(s+1+i.n)=  dxldt(s);
84          end
85                      % EB Liquid - Counter current HX
86                      % i.n = 10 -> Section 6
87          s = i.n/2+1;
88          dxldt(s) = (1/i.N_i_l)*(xl(i.n+2)-xl(s))*u(4)...
89                   +(p.U*p.Ai/(i.N_i_l*p.a_H2O1))*(xs(s)-xl(s));
90                      dxdt(s+2+i.n)=  dxldt(s);
91
92                      % EB Liquid - Counter current HX
93                      % i.n = 10 -> Section 2-5
94          for s = i.n/2:-1:2
95                  dxldt(s) = (1/i.N_i_l)*(xl(s+1)-xl(s))*u(4)...
96                   +(p.U*p.Ai/(i.N_i_l*p.a_H2O1))*(xs(s)-xl(s));
97                  dxdt(s+2+i.n)= dxldt(s);
98          end
99
100         dxdt = dxdt(:);
101         sys = dxdt;
102
103     case 2 % Discrete state udate
104
105         sys = []; %There is none
106
107     case 3 % Outputs
108
109         j= [];
110         for j=1:1:i.n*2+3
111             sys(j) = [x(j)];
112         end
113         sys = sys';
114
115     case 9 % Terminate
116
117         sys = [];
118
119     otherwise
120         error(['unhandled flag = ',num2str(flag)]);
121
122 end
```

A.6 Polyfit on enthalpy data

```
1 % Author : Bjørn Tore Mathisen
2 % Polynomial approximation of enthalpy data for 21-4-10 NPK
3
4 %Extracted data points:
5 clear all
```

```
close all
clf
%Polyfit approximation
x_temp = [25 32  34 34 40 42 54];          % Temperature
y_enth = [0 9.5 13 23 30 33 50];         % Enthalpy

figure(1)
hold on

   p = polyfit(x_temp,y_enth,5);
   x = [30:0.1:45];
   f = polyval(p,x);
   plot(x,f,'r')

   p = polyfit(x_temp,y_enth,4);
   x = [30:0.1:45];
   f = polyval(p,x);
   plot(x,f,'b')

   p = polyfit(x_temp,y_enth,3);
   x = [30:0.1:45];
   f = polyval(p,x);
   plot(x,f,'g')


   p = polyfit(x_temp,y_enth,2);
   x = [30:0.1:45];
   f = polyval(p,x);
   plot(x,f,'c')



   p = polyfit(x_temp,y_enth,1);
   x = [30:0.1:45];
   f = polyval(p,x);
   plot(x,f,'m')

   x_temp = [25  34 34  54];         % Temperature
y_enth = [0 13 23  50];
 plot(x_temp, y_enth,'k')
 axis([30 45 0 40])
 legend('5th deg', '4th deg', '3rd deg','2nd deg','1st deg')
  ylabel( 'Enthalpy [Kj/kg]')
  xlabel('T [C]')
```

A.7 Plot of phase change modelled over $\Delta T = 1$

```
clear all
close all
clf

```

```
%Testing
x_temp = [25  34 34  54];        % Temperature
y_enth = [0 13 23  50];         % Enthalpy
p.a_NPKbot = ((12-0)/(33.5-25)*1000);
p.a_NPKphase = ((24-12)/(34.5-33.5)*1000);
p.a_NPKtop = ((50-24)/(54-34.5)*1000);

  figure(1)
  hold on
  plot(x_temp, y_enth)
  plot([25 33.5 34.5 54],[0 12 24 50],'-r')
  ylabel( 'Enthalpy [Kj/kg]')
  title('Modelling cp over a Delta T = 1')
  xlabel('T [C]')
```

A.8 Evaluating trend data

```
%% 16-16-16(MOP)
% 11-11-2013 kl 0153

% Cp interpolation, from enthalpy Temperature diagram
% Linear approximation h = K_1 +aT
p.a_NPK1 = (32-9.5)/(42-32)*1000;   %J/kgK
p.a_H2O1 = 4183;                    %J/kgK

% Extracted process data
T_NPKin = 41.3; %C
T_NPKout= 33.3; %C
n_NPK = 208.0*(1000/(60*60)); %kg/s

T_wout= 19.6; %C
T_win = 32.5; %C

n_w = 58.5*(996.12/(60*60)); %kg/s

%Total Q comparisont
Q_NPK = n_NPK*p.a_NPK1*(T_NPKout-T_NPKin);
Q_w = -n_w*p.a_H2O1*(T_wout-T_win);

%Mean expirenced Cp
cp_NPK =-Q_w/(n_NPK*(T_NPKout-T_NPKin));

%% Presentation:
disp('------------ 16-16-16Cl November 11 ------------')
disp(' ')
disp('Total energy balance:')
text1 = ['Qwater: ',num2str(Q_w),' J/s'];
disp(text1)
text2 = ['QNPK: ',num2str(Q_NPK),' J/s'];
disp(text2)
disp(' ')
```

```
35 disp('Cp NPK calculated from Q_water:')
36 text1 = ['Cp_NPK: ',num2str(cp_NPK),' J/s'];
37 disp(text1)
38 text2 = ['p.a_NPK1: ',num2str(p.a_NPK1),' J/s'];
39 disp(text2)
40 disp(' ')
```

B.3 Operating environment screen shot

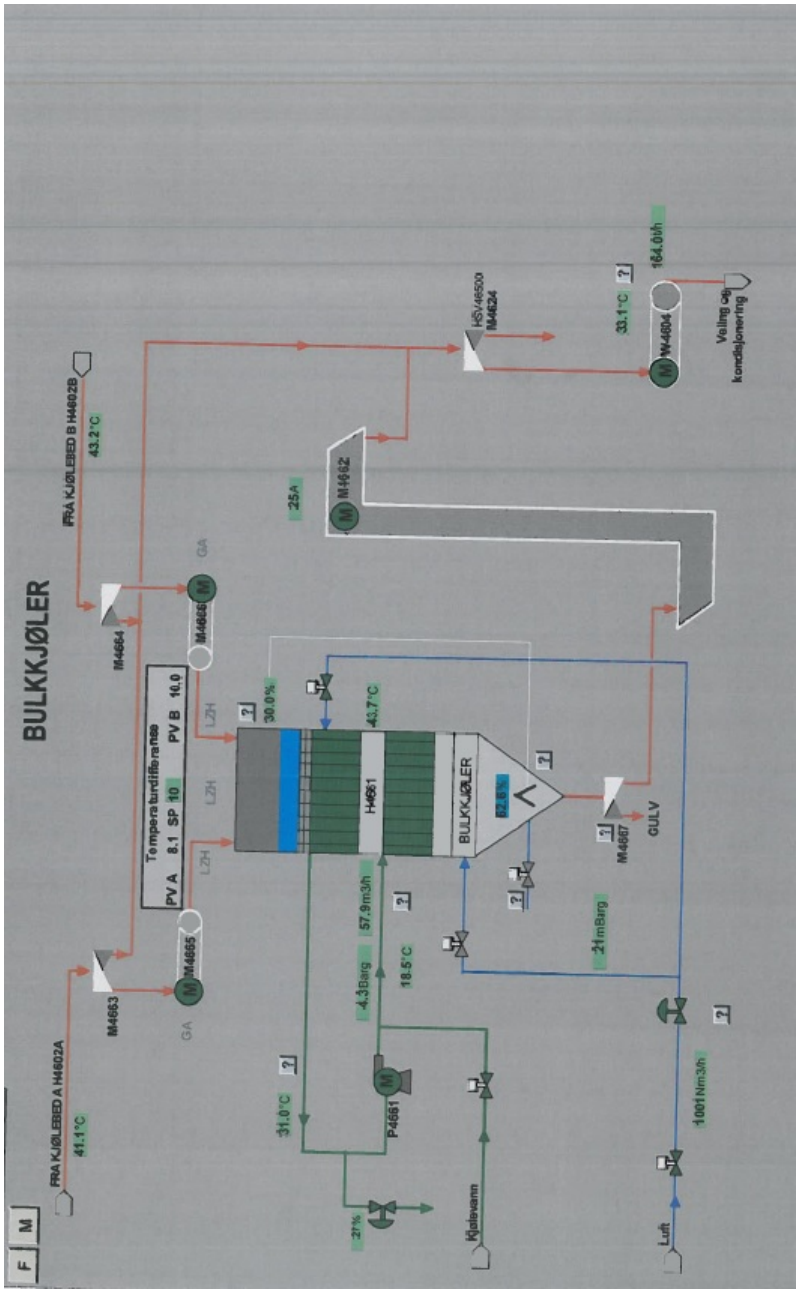
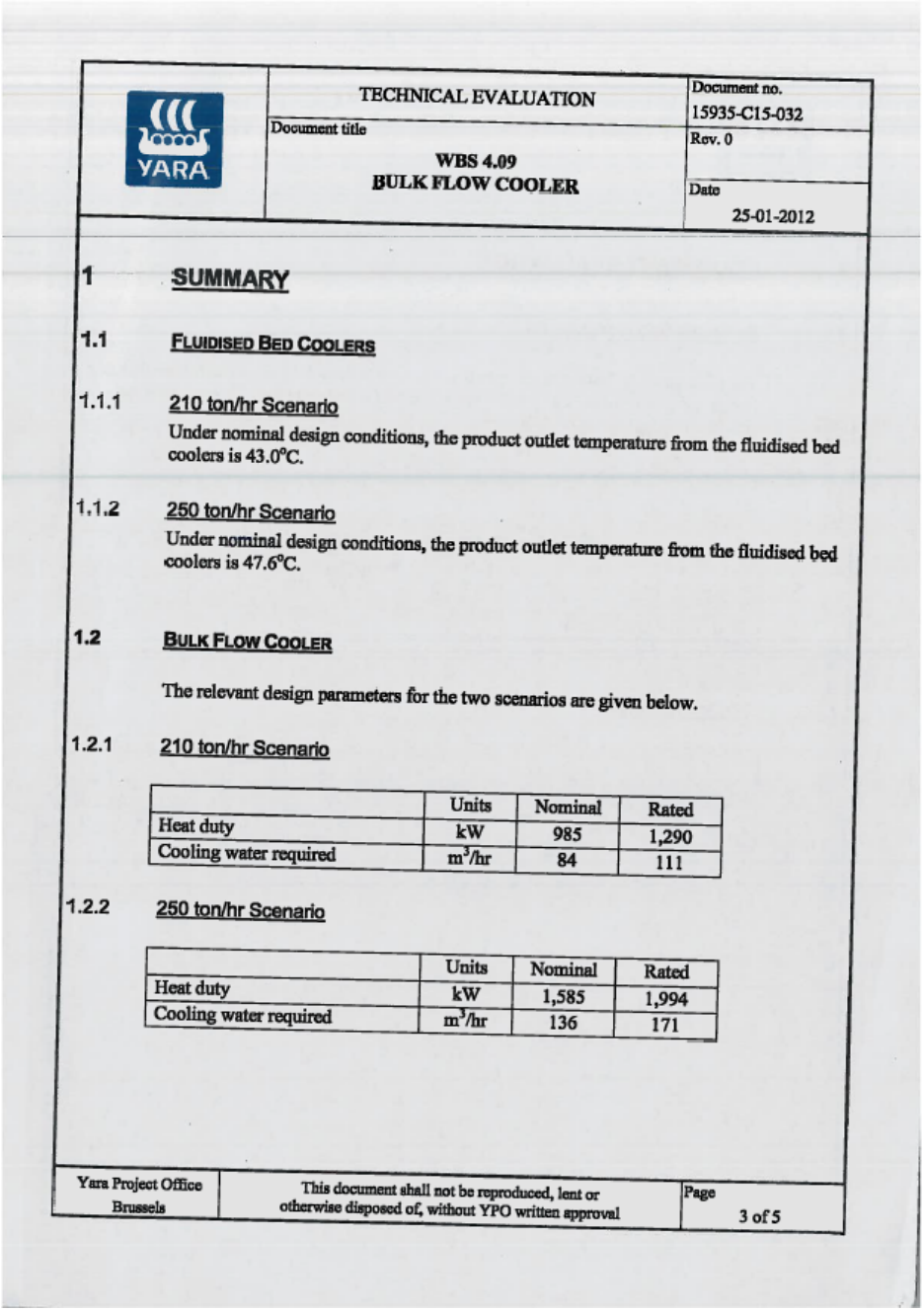


Figure 30: Screen shot of the operating environment

B.4 Technical Evaluation, WBS 4.09 Bulk flow cooler

B.4.1 Page 3



	TECHNICAL EVALUATION	Document no. 15935-C15-032
YARA	Document title: WBS 4.09 BULK FLOW COOLER	Rev. 0
		Date 25-01-2012

1 SUMMARY

1.1 FLUIDISED BED COOLERS

1.1.1 210 ton/hr Scenario

Under nominal design conditions, the product outlet temperature from the fluidised bed coolers is 43.0°C.

1.1.2 250 ton/hr Scenario

Under nominal design conditions, the product outlet temperature from the fluidised bed coolers is 47.6°C.

1.2 BULK FLOW COOLER

The relevant design parameters for the two scenarios are given below.

1.2.1 210 ton/hr Scenario

	Units	Nominal	Rated
Heat duty	kW	985	1,290
Cooling water required	m^3/hr	84	111

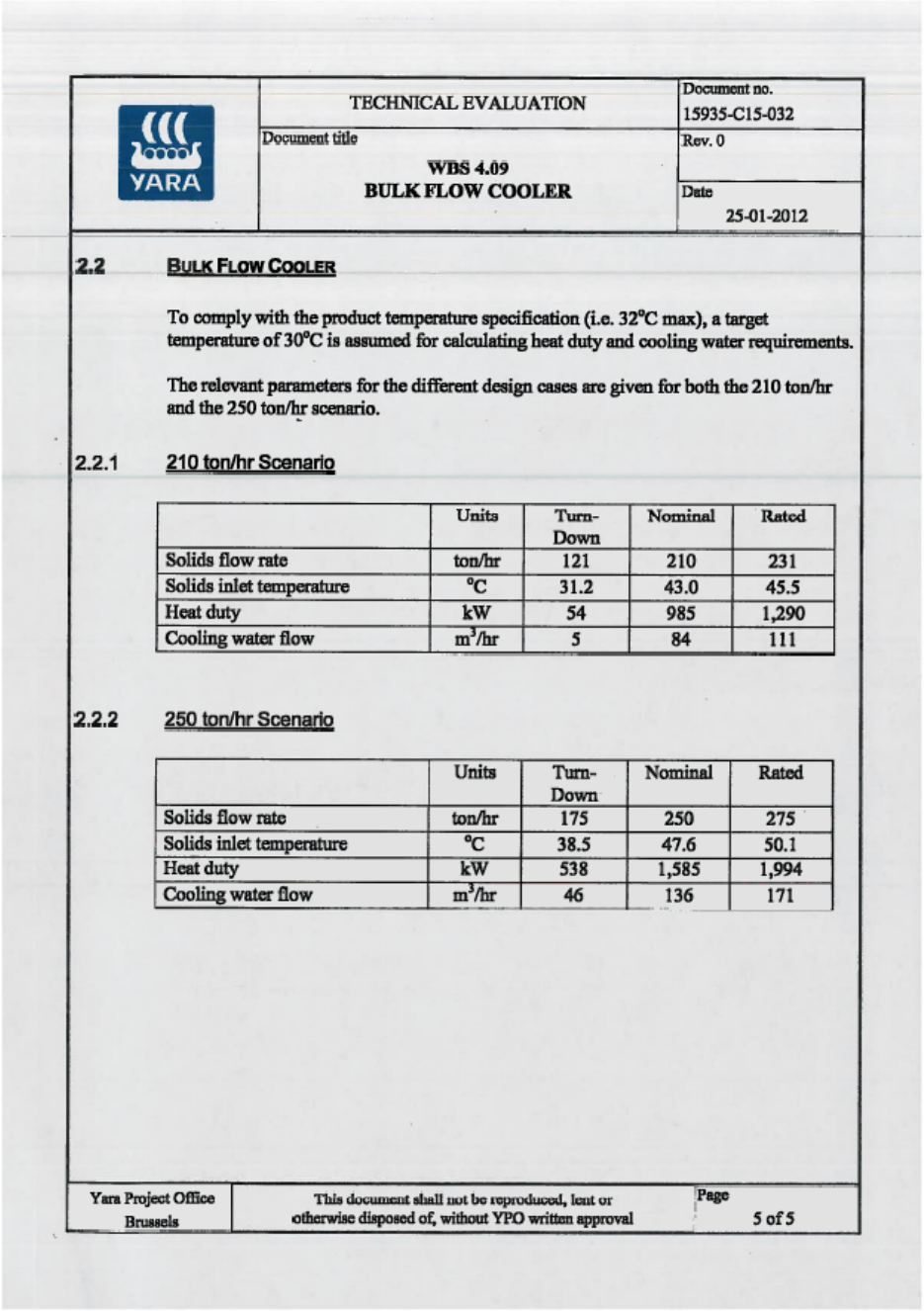
1.2.2 250 ton/hr Scenario

	Units	Nominal	Rated
Heat duty	kW	1,585	1,994
Cooling water required	m^3/hr	136	171

Yara Project Office Brussels	This document shall not be reproduced, lent or otherwise disposed of, without YPO written approval	Page 3 of 5

Figure 31: Technical evaluation by Yara, p.3

B.4.2 Page 5



TECHNICAL EVALUATION

Document no. 15935-C15-032

Document title: WBS 4.09 BULK FLOW COOLER

Rev. 0

Date 25-01-2012

2.2 BULK FLOW COOLER

To comply with the product temperature specification (i.e. 32°C max), a target temperature of 30°C is assumed for calculating heat duty and cooling water requirements.

The relevant parameters for the different design cases are given for both the 210 ton/hr and the 250 ton/hr scenario.

2.2.1 210 ton/hr Scenario

	Units	Turn-Down	Nominal	Rated
Solids flow rate	ton/hr	121	210	231
Solids inlet temperature	°C	31.2	43.0	45.5
Heat duty	kW	54	985	1,290
Cooling water flow	m^3/hr	5	84	111

2.2.2 250 ton/hr Scenario

	Units	Turn-Down	Nominal	Rated
Solids flow rate	ton/hr	175	250	275
Solids inlet temperature	°C	38.5	47.6	50.1
Heat duty	kW	538	1,585	1,994
Cooling water flow	m^3/hr	46	136	171

Yara Project Office Brussels

Page 5 of 5

Figure 32: Technical evaluation by Yara, p.5

B.5 Selected data for the 11th of November

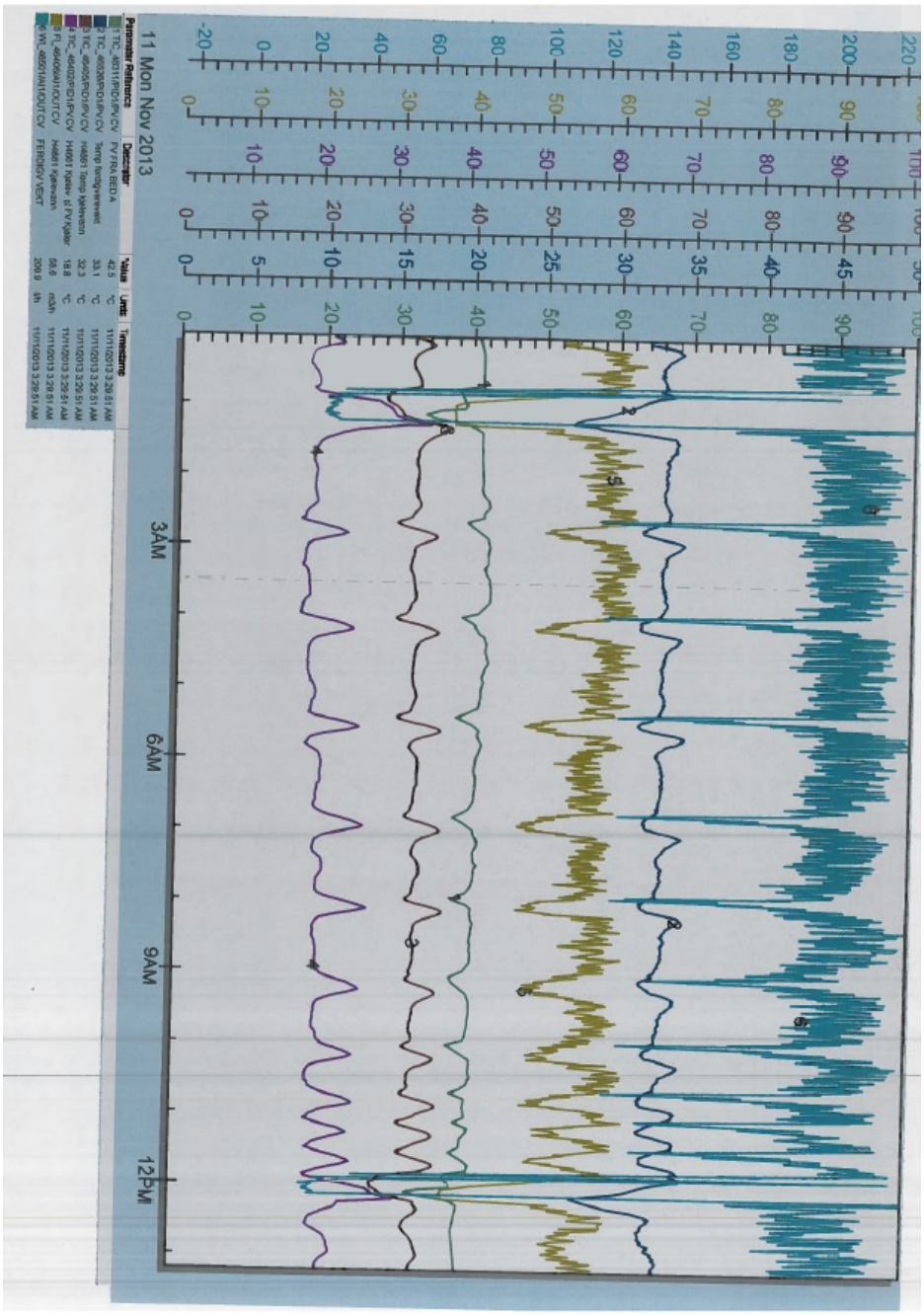


Figure 33: Selected data for the 11th of November environment

B.6 P& ID -Bulk Cooler

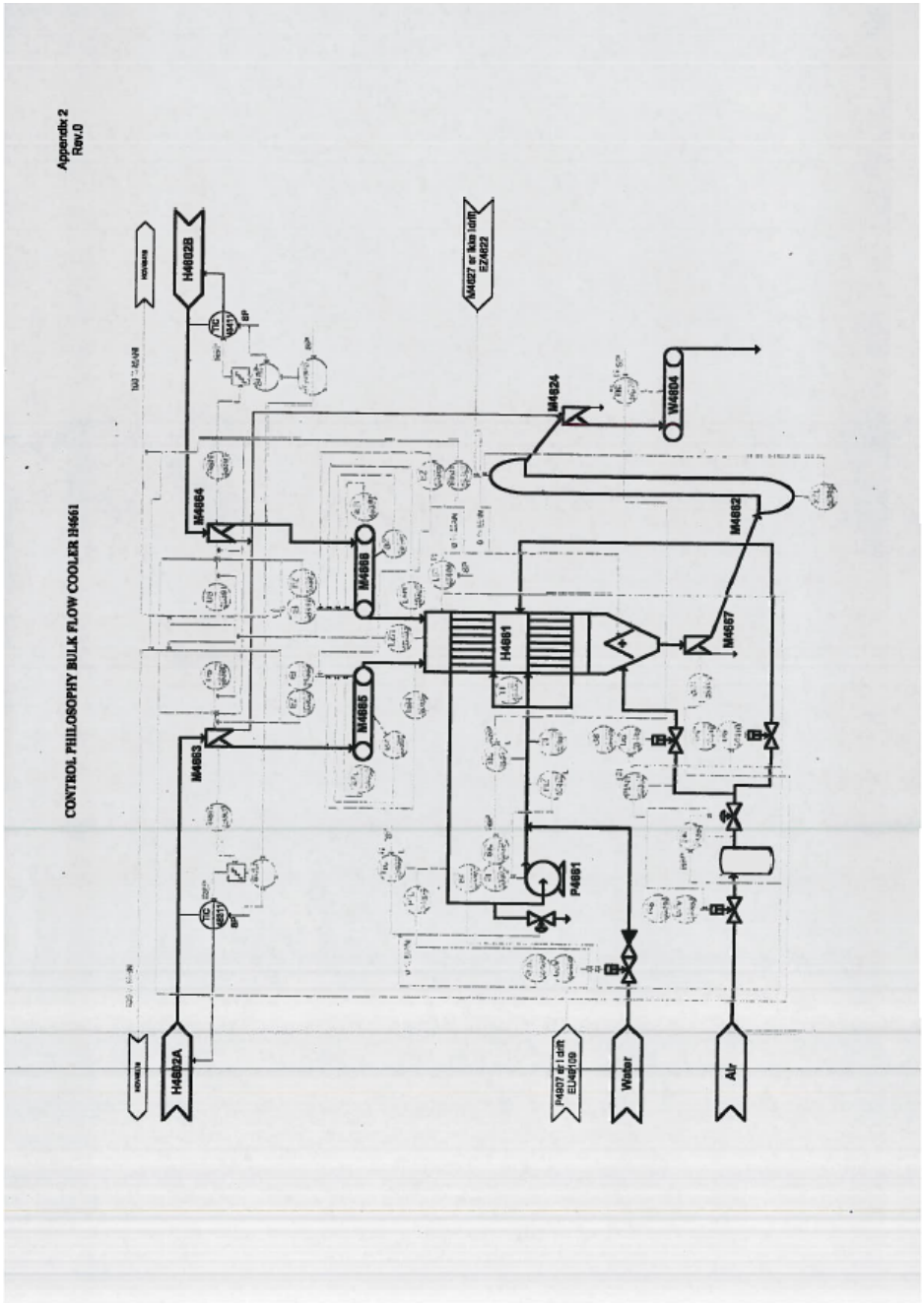


Figure 34: P& ID of the Bulk Cooler

B.7 Design specifications - Bulk Cooler

C Discarded models

C.1 Steady state model with a one dimensional distributed system

The model has the following topology: In this section the bulk cooler is

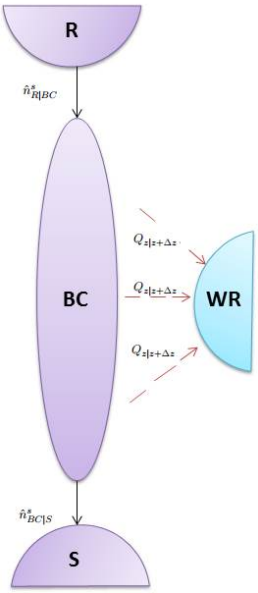


Figure 35: Topology of the bulk cooler modelled as a distributed system

simulated by utilizing knowledge about the fixed bed reactor [2]. For this early model only the steady state system is evaluated. The mass balance for the bulk cooler in it's entirety is then:

C.1.1 Mass balance

$$\frac{dN_B C^s}{dt} = \hat{n}^s_{R|BC} - \hat{n}^s_{BC|S} = 0 \tag{107}$$

C.1.2 Energy balance

For this model the air fertilizer mixture (air in the pockets between the prills) is considered homogeneous and the axial heat dispersion is neglected. The starting point for the energy balance is the total energy balance. Then the model lets the step in the axial direction from $z \to z + \Delta z$ approach zero. As mentioned earlier, only the steady state system is considered for this early

model with this approach. Taking the limit where the volume of an arbitrary section of the bulk cooler approaches zero :

$$\lim_{\Delta z \to 0} \frac{dH^s}{dt} = \lim_{z \to 0}(\hat{H}^s_{in} - \hat{H}^s_{out}) - \lim_{z \to 0} Q_{\Delta z} \tag{108}$$

$$0 = -\frac{\partial}{\partial z}\hat{H}^s - \frac{Ub\Delta z \Delta T}{\Delta z} \tag{109}$$

$$\hat{n}^s a_{NPK1} \frac{\partial T}{\partial z} = -Ub\Delta T \tag{110}$$

C.1.3 Parameter approximation

Calculation of heat exchanger area is done by taking the width of the heat exchanger and multiplying with 2 times the number of plates (each plate transfer heat on both sides) and the height of the heat transfer sections:

$$A_{Hx} = 2 \cdot 128 \cdot b_{BC} h_{Hx} \tag{111}$$

The total heat exchanger duty Q_{tot} is calculated by inserting design data for the overall energy balance for the liquid phase at steady state:

$$Q_{tot} = \hat{n}^l_{WR|D} a_{H2O1} \Delta T^l_{BC} \tag{112}$$

$$Q_{tot} = \frac{50000}{60} \cdot 4.183 \cdot (30 - 15)\ kJ/min \tag{113}$$

The heat transfer coefficient is then calculated by:

$$U = \frac{Q_{tot}}{\Delta T_{s-l}} \tag{114}$$

C.2 Steady state model with a one dimensional distributed system

C.2.1 Run file

```
%% Published by Bjorn Tore Mathisen
% Model of the Bulk Cooler in NPK 4 as part of a specialization
% subject H2013.
% The following model approximates the Bulk Cooler as a CSTR:

clc
clf
clear all
close all

% Process constants:
SSFconstant

```

```
% Global parameters
SSFparameters

% Interval in Z direction
zinterval = [0 p.h];

% Vector with initial conditons "x0".
x0 = [c.T_R];

% The initial conditions is used in the odesolver with reference to the
% function "equations" (see other script).
[z x] = ode23s(@Diff,zinterval,x0);

%%
% Graphical presentation of results:
figure(1)
plot(z,x(:,1))
title('Temperature')
xlabel('z [m]'); ylabel('T [C]');
xlim(zinterval)
```

C.2.2 Parameters and constants

```
%% Process constants for RunCSTRmod.m

global c i

% Gamma - process "constants"
c.T_R =   43;     % C Reservoir NPK temperature
c.T_cin = 18;     % C Temperature coolingwater in
c.T_cout= 30;     % C Temperature coolingwater out
c.T_cmid = (c.T_cout-c.T_cin)/2; % C Mean cooling water temperature
c.T_1  = 43;  % Celsius
c.n_R_BC= 210000/60; % kg/min
c.n_BC_S= 210000/60; % kg/min
c.n_WR_W= 50*1000/60 % kg/min ref 201 pr
```

```
%% Process parameters for RunCSTRmod.m

global p c i

% Cp interpolation, from enthalpy Temperature diagram
% Linear approximation h = K_1 +aT

p.a_NPK1 = (32-9.5)/(42-32);
p.a_H2O1 = 4.183;  %KJ/kgK

% Bulk cooler geomery
p.b = (1.7*2*128);% Hx width for each section
```

```
p.h = 4;           % Hx height
% NB 3 not confirmed height* width* plates
p.A = p.b*p.h;       %Total Hx area
p.U = ((50*1000/60)*4.183*15)/(p.A*(39-25));% QWater/(Ahx*dTHx)=U
```

C.2.3 Differential equations

```
% Differential equations for the 1-D distributed system
function dxdz = Diff(~,x0)
global c p

T = x0(1);
dTdz = 1/(c.n_R_BC*p.a_NPK1)*(-p.U*p.b*(T-c.T_cmid));

dxdz = [dTdz];
```

C.3 Steady state 2-D system

C.3.1 MATLAB script

To solve the system the explicit MATLAB solver ode23s was used. Belove follows an outline for the most significant lines of code. See appendix C.2 for full scripts.

```
% Process constants:
SSFconstant

% Global parameters
SSFparameters

% Interval in Z direction
zinterval = [0 p.h];

% Vector with initial conditons "x0".
x0 = [c.T_R];

% The initial conditions is used in the odesolver with reference to the
% function "equations" (see other script).
[z x] = ode23s(@Diff,zinterval,x0);
```

Where the differential equations are stored in the function Diff:

```
% Differential equations for the 1-D distributed system
function dxdz = Diff(~,x0)
global c p

T = x0(1);
```

```
dTdz = 1/(c.n_R_BC*p.a_NPK1)*(-p.U*p.b*(T-c.T_cmid));

dxdz = [dTdz];
```

C.4 2-D model

```
%% Published by Bjorn Tore Mathisen
% Model of the Bulk Cooler in NPK 4 as part of a specialization
% subject H2013.
% The following model approximates the Bulk Cooler as a CSTR:

clc
clf
clear all
close all

global y p

disp('Initate')
% Process constants:
SSF2constant

% Global parameters
SS2Fparameters

% Interval in Z direction
zinterval = [0 p.h];

% Discretizing in the y direction
y = 0:p.whalfsec/(p.yp-1):p.whalfsec;

% Vector with initial conditons "x0".
x0 = c.T_R*ones(p.yp,1);

% Remove end points from being treated ass ODE's. Thes are
% strictly algebraic equations.
M = eye(p.yp);          % Single matrix for every variable
M(1,1) = 0.0;
M(end,end) = 0.0;
M = sparse(M);

options = odeset('Mass',M,'Stats','on','RelTol',0.001,'AbsTol',0.001);

%Calling ODE function, integrates over the length heat exchanger - z
[z,x] = ode15s(@Diff2,zinterval,x0,options);

%%
% Graphical presentation of results:
figure(1)
surf(y,z,x)
title('Temperature')
```

```
46 xlabel('y [m]'); ylabel('z [m]'); zlabel('T [C]')
```

```
1  % Differential equations for the 2-D distributed system
2  function dxdz = Diff(~,x0)
3  global c p y
4
5  T = x0;
6
7  m  = length(y);
8  y1 = y(1);
9
10 %Calculating the first derivative of T
11 dTdy    = dss020(y1,y(m),m,T',1);
12
13 %Calculating the second derivative of T
14 d2Tdy2  = dss042(y1,y(m),m,T',dTdy,2,2);
15
16 %Boundary condition for first derivative in T
17 res1T   = dTdy(1);
18 disp(T(p.yp))
19 disp(dTdy(p.yp))
20 disp((p.U/c.lambda*(T(p.yp)-c.T_cmid))/10000)
21 %Boundary condition for second derivative in T
22 res2T   = -dTdy(p.yp) -(p.U/c.lambda*(T(p.yp)-c.T_cmid)/10000)
23
24 dTdz = 1./((c.n_R_BC./129).*p.a_NPK1)*(c.lambda.*(d2Tdy2'));
25 dxdz = [res1T; dTdz(2:p.yp-1); res2T ];
```

C.4.1 Mass matrix

To effectively evaluate the differential with implementation of boundary conditions the mass matrix is introduced in the ode solver options. The mass matrix (Discontinued as CSTR in series was the most promising model)

D Attempt at modelling in Aspen

A large portion of the available work time was used in trying to model the system in Aspen plus 8.2. Or rather trying to make Aspen 8.2 work on authors computer. Aspen plus 8.2 was updated, re-installed with the NTNU license and re-installed again using Yara's Aspen plus copies. All attempts failed as the following crashes re-appered after each re-install.

D.1 Application crashes

D.1.1 The "Help" crash

The help crash was instantaneous and 100% "reliable"

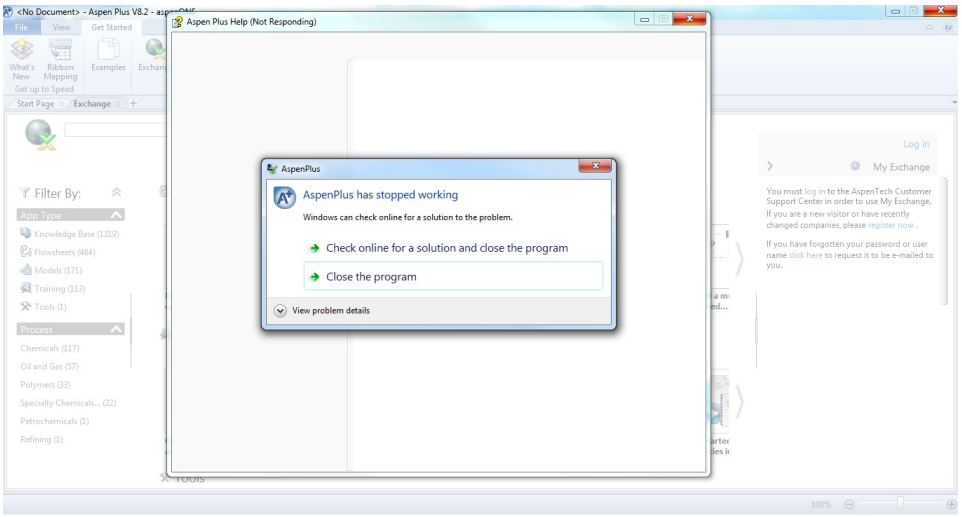


Figure 36: When pressing the help button in the top right corner Aspen plus 8.2 instantaneously crashed

D.1.2 The "Training" crash

The training crash was also instantaneous and 100% "reliable"

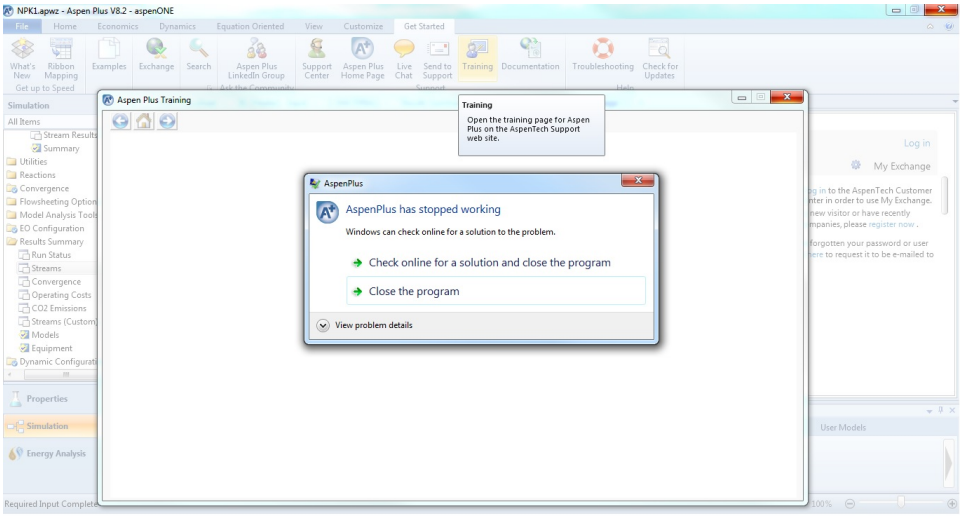


Figure 37: When pressing the "training button" in the middle of the bar Aspen plus 8.2 instantaneously crashed

D.1.3 The start up crash

This crash was a bit more random. Happening approximately 40% of the start up attempts:

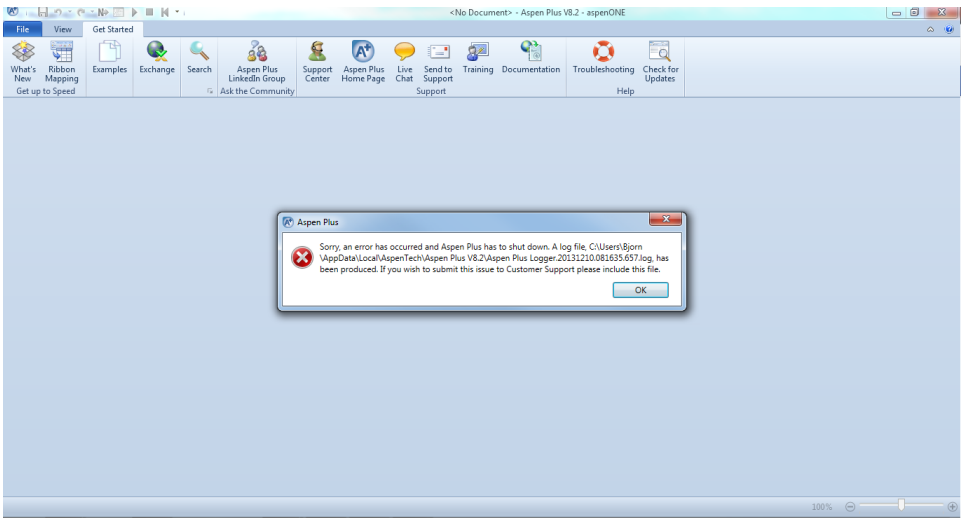


Figure 38: Aspen Plus 8.2 had several crashes when attempting to start the program

D.1.4 General application crashes

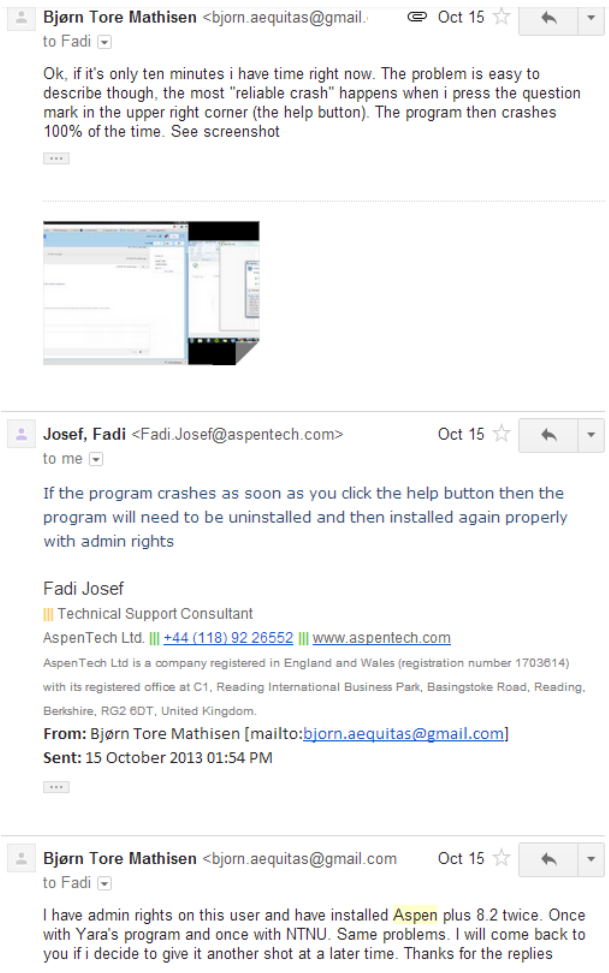
The rest of the crashes was a bit more unpredictive. But on october 8 the project was cancelled after a crash occurred after inserting a predicted molar weight for NPK and restarting did not help. The following is small out-print of the crash log. If interested the entire crash log can be delivered up on request to *bjorntma@stud.ntnu.no*

```
1  [01:31:43]:Unhandled Exception
2  Exception Message: Input string was not in a correct format.
3  Excetion Source: mscorlib
4  StackTrace:    at System.Number.ParseDouble(String value, NumberStyles options, NumberFormatInfo nu
5     at AspenTech.AspenPlus.WpfFindComponent.UserDefinedCompWizardMain.WriteNodeParameters()
6     at AspenTech.AspenPlus.WpfFindComponent.UserDefinedComp1.OnForward(WizardControlItem targetItem)
7     at AspenTech.AspenPlus.ModelLibrary.Controls.WizardControl.GoForward()
```

Figure 39: A crash occuring trying to manually assign user defined substance "NPK" molecular weight

D.1.5 Technical support correspondance

I had a short correspondance with tech support. It was short because they recommended re-installed after i had done it a number of times and because I had to achieve some progress on my project and decided together with my supervisors to turn to MATLAB for the modelling:



Bjørn Tore Mathisen <bjorn.aequitas@gmail.
Oct 15
to Fadi

Ok, if it's only ten minutes i have time right now. The problem is easy to describe though, the most "reliable crash" happens when i press the question mark in the upper right corner (the help button). The program then crashes 100% of the time. See screenshot

Josef, Fadi <Fadi.Josef@aspentech.com>
Oct 15
to me

If the program crashes as soon as you click the help button then the program will need to be uninstalled and then installed again properly with admin rights

Fadi Josef
Technical Support Consultant
AspenTech Ltd. +44 (118) 92 26552 www.aspentech.com
AspenTech Ltd is a company registered in England and Wales (registration number 1703614) with its registered office at C1, Reading International Business Park, Basingstoke Road, Reading, Berkshire, RG2 6DT, United Kingdom.
From: Bjørn Tore Mathisen [mailto:bjorn.aequitas@gmail.com]
Sent: 15 October 2013 01:54 PM

Bjørn Tore Mathisen <bjorn.aequitas@gmail.com
Oct 15
to Fadi

I have admin rights on this user and have installed Aspen plus 8.2 twice. Once with Yara's program and once with NTNU. Same problems. I will come back to you if i decide to give it another shot at a later time. Thanks for the replies

Figure 40: Correspondence with aspen tech support