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

SPECIALIZATION PROJECT 2015

TKP 4555

PROJECT TITLE:

Robust implementation of optimal operation of LNG refrigeration cycles

By

Alexander Leguizamón Robayo

Supervisor for the project: Sigurd Skogestad
Co-supervisor for the project: Adriana Reyes Lúa

Date: 18.12.2015

Robust implementation of optimal operation of LNG refrigeration cycles

Alexander Leguizamon

December 17, 2015

Abstract

A simulation model was built for the on-board liquefaction process proposed by Neksået al. (2010). An operation degrees of freedom analysis was carried out for the system. In this project the modelling approach calculates each unit independently in Matlab. This model is proposed having in mind its future optimization. For this purpose the model requires to be robust and reliable. The properties of the stream are calculated based on Soave's modification of the Redlich-Kwong equation of state (Soave, 1972). Different approaches for the flash calculations are discussed based on their robustness and complexity. The reliability of the model is evaluated by comparing the results using a model developed on the commercial process simulation software Aspen Hysys®.

Acknowledgements

I like to acknowledge Adriana Reyes for her time and support during the development of this project. I would like to thank her specially for all the motivation and feedback throughout this semester.

I would also like to thank Sigurd Skogestad for this allowing me to work on on this project as a part of the Process Systems Engineering group.

List of Figures

List of Tables

List of symbols

α	Alpha parameter SRK
α_i	Alpha parameter SRK for an i component
η	Compressor efficiency
ω	Acentric factor
ω_i	Acentric factor for an i component
$\overline{C_p}$	Weighted heat capacity at average temperature in the compressor
$\phi_{i,l}$	Liquid fugacity coefficient for an i component
$\phi_{i,v}$	Vapour fugacity coefficient for an i component
A	Heat exchanger area
A	A parameter SRK
$A_{i,j}$	Interaction parameter for an i and j component pair
A_i	A parameter for heat capacity polynomial for an i component
$A_{p,i}$	Partial A parameter SRK for an i component
B	B parameter SRK
B_i	B parameter for heat capacity polynomial for an i component
$B_{p,i}$	Partial B parameter SRK for an i component
C_i	Correction for the fugacity for an i component
C_i	C parameter for heat capacity polynomial for an i component
$C_{p,av}$	Heat capacity at average temperature in the compressor
$C_{p,in}$	Heat capacity evaluated at the inlet of the compressor
$C_{p,i}$	Heat capacity for an i component
D_i	D parameter for heat capacity polynomial for an i component
$Error_{liquid}$	Error on the compressibility for liquid phase

$Error_{vapour}$	Error on the compressibility for vapour phase
f	Fugacity
f_i	Fugacity for an i component
h_f	Molar enthalpy for the feed stream
h_l	Molar enthalpy for the liquid stream
h_v	Molar enthalpy for the vapour stream
$h_{ideal,i}$	Ideal molar enthalpy of an i component
h_{ideal}	Ideal molar enthalpy
h_{in}	Inlet molar enthalpy for a stream entering a unit
h_{out}	Outlet molar enthalpy for a stream exiting a unit
K	Vapour-liquid distribution ratio
K_i	Vapour-liquid distribution ratio for an i component
$LMTD$	Logarithmic mean of the temperature difference
m	Slope as a function of the acentric factor
m_i	Slope as a function of the acentric factor for an i component
N	Molar flow
N_{SS}^{max}	Maximum steady state degrees of freedom
N_i	Molar flow for a stream i
N_0	No steady effect degrees of freedom
N_{MV}	Manipulated variables degrees of freedom
N_{SS}	Steady state degrees of freedom
P	Pressure
P_c	Critical pressure
$P_{c,i}$	Critical pressure for an i component
P_{in}	Inlet pressure for a stream entering a unit
P_{out}	Outlet pressure for a stream exiting a unit
$P_{Re,i}$	Reduced pressure for an i component
P_{Re}	Reduced pressure
Q	Heat duty for heat exchanger
Q_i	Heat duty for heat exchanger i

R	Universal gas constant
$SRK(T, P, z, phase)$	Calculation of the compressibility for given conditions algorithm 3.2
T	Temperature
T_c	Critical temperature
T_{av}	Average temperature for a stream in a given unit
$T_{c,i}$	Critical temperature for an i component
T_{in}	Inlet temperature for a stream entering a unit
$T_{out,new}$	Last value for the outlet temperature for a stream exiting a unit
$T_{out,old}$	Previous iteration value for the outlet temperature for a stream exiting a unit
T_{out}	Outlet temperature for a stream exiting a unit
$T_{Re,i}$	Reduced temperature for an i component
T_{ref}	Reference temperature
T_{Re}	Reduced temperature
U	Overall transfer coefficient
v_f	Vapour fraction
W_C	Compressor work
x	Liquid phase composition
x_i	Liquid phase composition for an i component
x_i	Mole fraction of an i component(For equation 3.5)
y	Vapour phase composition
y_i	Vapour phase composition for an i component
Z	Compressibility factor
z	Feed composition
z_i	Feed composition for an i component
Z_l	Compressibility for the liquid phase
Z_v	Compressibility for the vapour phase

Contents

Chapter 1

Introduction

During the last years, liquefied natural gas (LNG) has been one of the fastest growing energy sources and it is expected to increase its share in the energy market in the future years. The liquefaction process allows large volumes of natural gas to be transferred through long distances. However, this procedure requires energy demanding cooling through a broad range of temperatures, specially at low temperatures. Therefore, research on this area has increased and it has lead to the use and proposition of many process concepts, so that the requirements of small temperature driving forces are met (Austbø, 2015).
There has been an increasing interest in remote was production and floating operations. This has brought up new challenges for the LNG processes. One of the challenges that arises is the transportation of LNG to remote places. As a response to this situation, small vessels are used to deliver LNG to areas in which natural gas is not available.
During the transportation of LNG in small ships, due to the changing environment conditions, there is a fraction of LNG that vaporizes. This vapour is known as boil-off gas (BOG) and it can become a representative product loss. Hence, in order to reduce these losses, an on-board alternative, known as mini-LNG (Figure 2.1), has been proposed by Neksået al. (2010).
Small units have to be specially cost and energy efficient (Neksået al., 2010). As the design is already proposed, the next step is to optimize the operation of the scheme. Along these lines, surges then the need for a model of the mini-LNG plant. This model has to be:

- Reliable: So that the model accurately describes the behaviour of the actual plant.
- Robust: In order to assure convergence of the model through the optimization routine

This project aims to propose a model that meets the mentioned requirements. For that objective, the strategy carried out involves decisions related to the modelling approach, thermodynamic modelling, models for the units and specific process assumptions among others.

Chapter 2

Process Description

2.1 Process Description

One of the plans of the LNG industry in the near future is to distribute LNG to remote areas by using small carriers. In other to minimize losses, these small carriers count with technology to reliquify the LNG Boil-off gas. This is achieved by the mini-LNG configuration proposed by Neksået al. (2010).

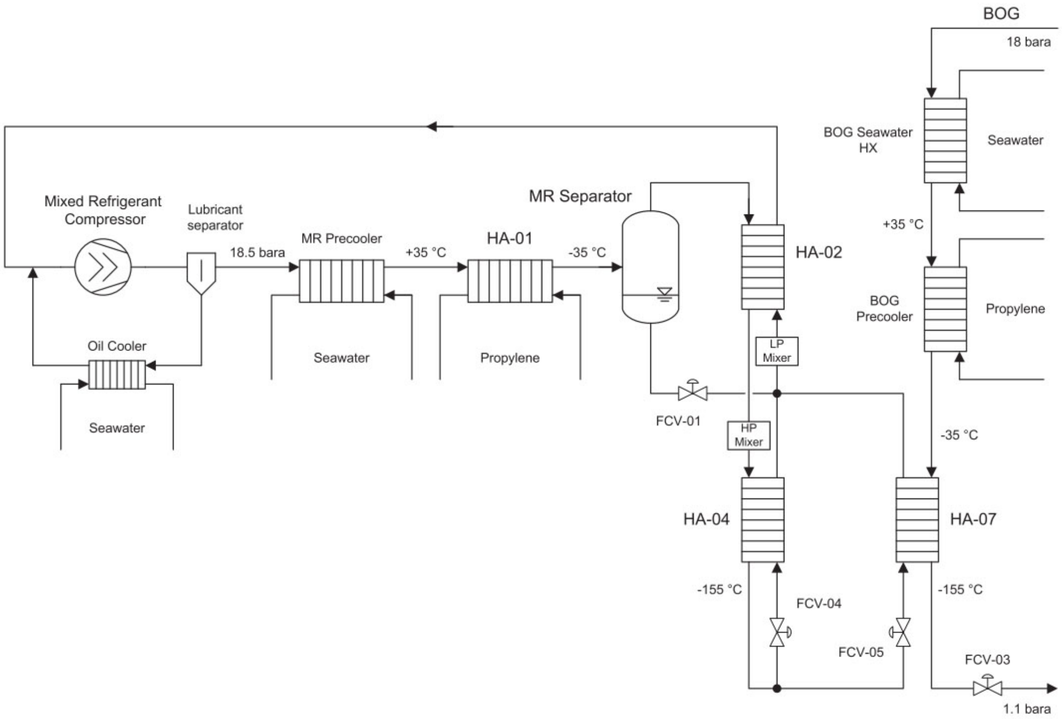


Figure 2.1: Mini LNG Configuration (Neksået al., 2010)

The proposed configuration shown on Figure 2.1 counts with several plate heat exchangers which only allow two parallel flows. In this process the low pressure refrigerant condenses and sub-cools the high pressure refrigerant through the units HA-02 and HA-04. On the other hand, the natural gas is precooleded and desuperheated before entering to the unit HA-07, in which the gas will be condensed and subcooled. The low pressure refrigerant to be compressed in the mixed refrigerant compressor. After the lubricant has been removed the refrigerant is precooled by sea water, then further cooled by the refrigeration plant HA-01.

The cooled refrigerant is separated in the MR separator, from which two streams come out. One stream being in vapour phase with a higher concentration of the volatile components and the other being the liquid with a higher fraction of the heavier components. The liquid passes then through the valve FCV-01, in which its pressure is reduced. This reduced pressure liquid will be mixed with the refrigerant coming out of HA-07 and will be used to cool the vapour stream on unit HA-02. The high pressure vapour will pass then through unit HA-04 where it is further condensed and subcooled. This stream is later throttled by FCV-04 and FCV-05 in order to work as a coolant for the two coldest units of the system (HA-04 and HA-07).

The boil-off gas (BOG) comes into the system at high pressure, then it is precooled in the sea water heat exchanger, further cooled by the refrigeration plant then it is desuperheated, liquefied and subcooled in HA-07. The subcooled gas is finally throttled to the storage tank pressure by FCV-03.

There are some considerations to be made in this scheme:

- The amount of lubricant present in the process after the separation is negligible.
- The temperature difference between the cold evaporating refrigerant and the hot streams can be adjusted by modifying the composition of the refrigerant.
- In practice, this process is subject to limitations due to the size of the equipment and their maximum operating pressures. The maximum design operating pressure for the heat exchangers is 31 bara. For the BOG the limitations is assumed to be 22 bara due to the compressor pressure limitation.

The composition of the refrigerant is shown in Table 2.1. As for the BOG, its composition is shown in table 2.2.

Table 2.1: Refrigerant composition used for analysis

Component	Mole fraction
Nitrogen	0.1074
Methane	0.3292
Ethane	0.4096
Propane	0.1345
Butane	0.0193

Table 2.2: Boil Off Gas Composition

Component	Mole fraction
Nitrogen	0.11
Methane	0.89

In order to start modelling this process, Figure 2.2 shows the values for pressures and temperatures for the most relevant streams in the process. These values were used as a starting point to

model the process.
It should be noted that further assumptions will be considered in order to simplify the model.

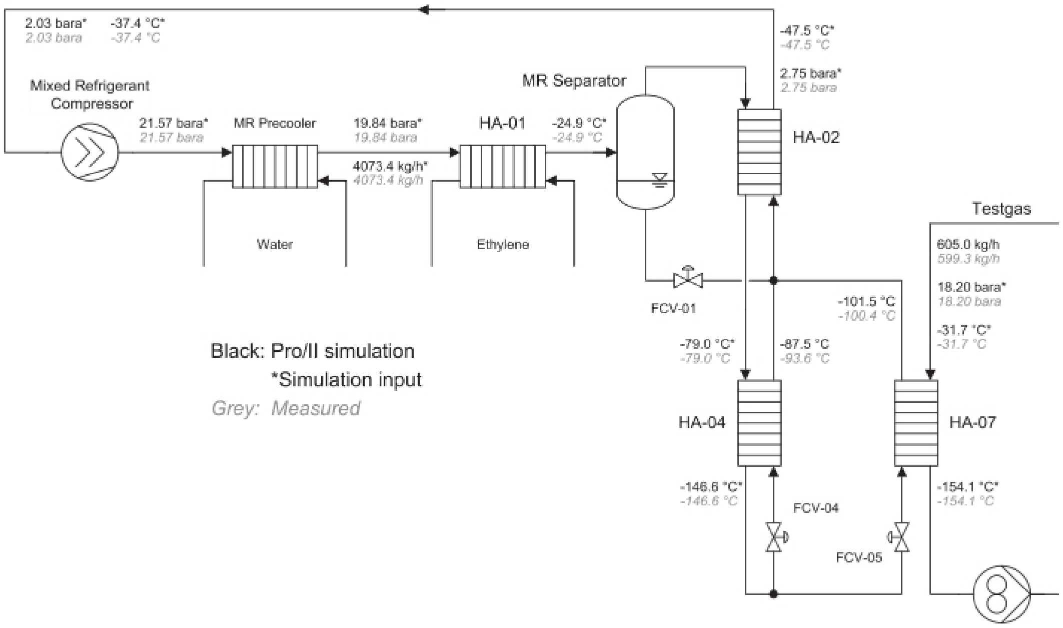


Figure 2.2: Mini LNG configuration with parameters values (Neksået al., 2010)

2.2 Problem Description

As explained on section 1 this model has to be defined in such a way that it suits the optimization requirements. These requirements correspond to reliability and robustness.
In order to carry out this task the problem has been structured in different parts. The first is a preliminary analysis in which more specific assumptions for the model will be considered and an optimization degrees of freedom analysis will be carried out in order to have an initial scope for the parameters, which are necessary to know for the steps. The following part is a modular sequential simulation, in which each of the units will be modelled independently; this first step will not only lead to a good initial point for a entirely equations oriented model, but also make it easier to evaluate the validity of the models for each unit. The final step is to propose improvements for the robustness of the model so that it is more suited for optimization. The following section(2.2.1 will explore this approach in detail and the thought process behind it.

2.2.1 General Modelling Decisions

Approach

As mentioned earlier this model is intended for optimization. For this purpose there are two main approaches: The sequential modular (SeqM) and the equations oriented (EO).

For the SeqM case, all the units are calculated in sequence for each iteration. Using this approach, the first challenge that arises is to have a simulation that converges every time it is called by the optimization routine (Jacobsen and Skogestad, 2011). However, there are some situations in which the convergence cannot be guaranteed for all iterations. This problem can be circumvented by using infeasible path algorithms such as the one proposed by Biegler (1985) and Biegler and Cuthrell (1985). The core principle of this algorithm is to use the optimization routine to take care of the convergence of the flowsheet instead of the process simulator (Austbø, 2015).

The other approach corresponds to the EO, in which all the equations of the model are solved simultaneously. In this way all (or most of) the equations are explicit, thus the exact derivatives can be calculated directly, however this requires a more careful initialization and problem formulation, in addition to the use of a more robust solver (Kamath et al., 2010).

Given the kind of plant that is modelled and its complexity (Figure 2.1), it is possible to notice that the flowsheet involves several interacting recycles. Based on this, it is preferably to use an EO model for the optimization (Morton, 2003).

As this model is made from scratch, the decided approach is to implement each unit operation independently in order to make sure that each unit is modelled correctly. The objective of this work is to obtain a model that works and can provide an appropriate initialization point for a future EO model, in which the optimization will be carried out.

Modelling Software

After a modelling approach has been set, it is necessary to make a decision regarding the software to use for the model. There are many alternatives available. Most of the LNG simulations have used commercial modular sequential alternatives, such as Aspen Hysys®, Aspen Plus®, Honeywell UniSim® or PRO/II. The strongest advantage that brings the use of these simulators is the simple modelling interface and the fact that the simulator takes care of the mass and energy balances constrains (Austbø, 2015).

In addition to these alternatives, it is possible to carry out the model using mathematical computing software such as Matlab. The use of these alternatives has the main advantage of being able to deeply understand and manipulate the models by having all the equations and assumptions available, additionally Matlab counts with much more robust optimization algorithms than those in the commercial process simulators.

Some other works such as Sharifzadeh and Thornhill (2011), Jacobsen and Skogestad (2011) and Helgestad (2010), have used both a commercial process simulator (Aspen Hysys®) for the modelling of the LNG unit and Matlab to carry out the optimization. This approach takes advantage of the robustness and reliability of the Matlab optimization algorithms while keeping the perks of using a commercial simulator. Nonetheless, there is still the disadvantage of not being able to calculate the derivatives using a commercial simulator, these can lead to problems in the optimization results as seen in Helgestad (2010) work. Finally, the software chosen to model this process is Matlab, given the advantages previously mentioned and the possibility of further work into a full EO model. However, Aspen Hysys® will be used as a tool to validate the modelling results, as it has been done by Michelsen et al. (2010).

Chapter 3

System Model

3.1 Degrees of Freedom Analysis

As the whole system is going to be optimized, it is useful to perform a degrees of freedom (DOF) analysis before carrying out the modelling of the system. It should be noted that this is a DOF analysis for optimization which is different to a design DOF analysis. This analysis is based on the previous studies carried out by Jensen and Skogestad (2009).
The general formula for the steady state degrees of freedom (N_{SS}) is:

$$N_{SS} = N_{MV} - N_0 \tag{3.1}$$

where N_{MV} corresponds to the manipulated variables and N_0 to the manipulated variables with no steady-state on the operation and variables without an steady-state effect that are required to be controlled, such as liquid hold ups.
Based on figure 2.2, it is possible to find the maximum possible degrees of freedom. They correspond to all the possible manipulated variables in the process, according to Jensen and Skogestad (2009), there is one possible DOF for each valve and compressor. For heat exchangers there is at least one DOF, assuming that one flow can be controlled; in the case where there is a bypass, there is one additional DOF per bypass. Additionally, on an ideal scenario the composition of the refrigerant could be adjusted, thus giving $NC-1$ extra DOF.This calculation is shown on table 3.1

Table 3.1: Maximum DOF

1	Compressor
5	Heat Exchangers
3	Valves
4	Compositions
13	N_{SS}^{max}

Considering the configuration of the system the following DOFs are removed: Two heat exchangers without bypass (HA-02 and HA-04), the heat exchangers (MR Precooler and HA-01) are taken as one (The reasoning behind this assumption is discussed in the following section) and due to the nature of the process, the composition of the refrigerant will remain constant. By calculating the difference between the maximum possible DOF and the lost DOF, it is possible to know the number of actual manipulated DOF (N_{MV}), as shown in table 3.2. The final step now

Table 3.2: Lost DOF

-2	No by pass(HA-02 and HA-04)
-1	Precooler + HA-01
-4	Fixed compositions
-7	Lost DOF
13	N_{SS}^{max}
6	N_{MV}

is to calculate the actual DOF, using equation 3.1. For this case N_0 is only one and corresponds to a valve controlling the level in the flash tank.

Table 3.3: Actual DOF

6	N_{MV}
-1	N_0
5	N_{SS}

These last five DOF (Figure 3.1) correspond to:

- 1 Compressor work
- 2 Valves (FCV-04, FCV-05)
- 2 Heat duties (MR Precooler + HA-01, HA-07)

It should be noted in this section that the output pressure for both valves is the same. So one of the valves will be used to set the pressure and the other will be used to control the flow ratios. Further studies in this regard are recommended for future work.
Regardless of the objective function, of which definition is out of the scope of this project, there are two clear sources of disturbances:

- Between the output of the unit HA-02 and the compressor, there is a considerable heat transfer between the environment and the refrigerant. This value is by no means possible to control and can be one of the main perturbations used for future optimization studies
- The conditions of the BOG stream. These can change depending on the environment conditions. Additionally, depending on the optimization problem, the flow of this stream can either be a DOF or a perturbation. This last issue is further discussed on section 5.1.

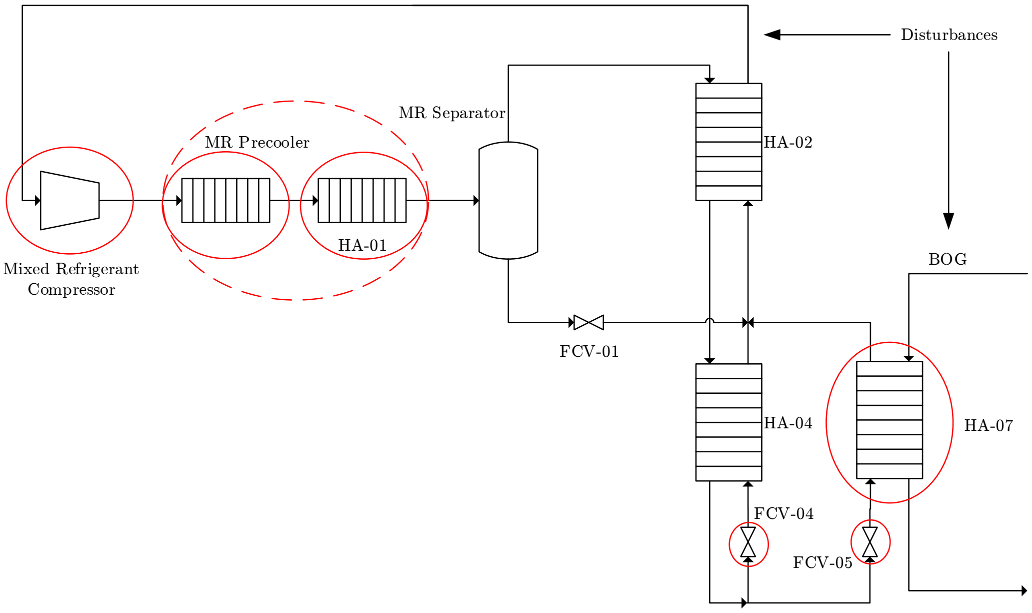


Figure 3.1: Mini LNG configuration with highlighted degrees of freedom

3.2 Thermodynamic Model

This section contains information of the equations used to estimate the properties of the different mixtures present in the process. The implementation of the following equations and the equipment has been validated comparing to an Aspen Hysys® simulation. Further discussion in that regard is carried out on section 4.3.

3.2.1 Equation of State & Multicomponent Model

The thermodynamic modelling for this system is based on the modification of Redlich-Kwong equation of state proposed by Soave Soave (1972) (also known as SRK). This equation has been selected due to its reliability to describe LNG systems (Austbø, 2015). This equation allows to find the compressibility of a a component for a given temperature and pressure.

SRK model for a pure component:

$$T_{Re} = T/T_c \tag{3.2a}$$

$$P_{Re} = P/P_c \tag{3.2b}$$

$$m = 0.480 + 1.574 * \omega - 0.176 * \omega^2 \tag{3.2c}$$

$$\alpha = (1 + m * (1 - T_{Re}^{0.5}))^2 \tag{3.2d}$$

$$A = 0.42747 * \alpha * P_{Re}/T_{Re}^2 \tag{3.2e}$$

$$B = 0.08664 * P_{Re}/T_{Re} \tag{3.2f}$$

$$Z^3 - Z^2 + Z(A - B - B^2) - AB = 0 \tag{3.2g}$$

$$ln\left(\frac{f}{P}\right) = Z - 1 - ln(Z - B) - \frac{A}{B} ln\left(\frac{Z + B}{Z}\right) \tag{3.2h}$$

Where: T is the temperature, T_{Re} is the reduced temperature, T_c is the critical temperature, P is the pressure, P_{Re} is the reduced pressure, P_c is the critical pressure, ω is the acentric factor, Z is the compressibility factor and f is the fugacity; the rest are parameters of the model. However in this system there are not streams which have only one component, thus it is necessary to modify the model(3.2) for a mixture. Including the mixing rules, equation 3.2 becomes: SRK model for multiple component:

For a component i:

$$T_{Re,i} = T/T_{c,i} \tag{3.3a}$$

$$P_{Re,i} = P/P_{c,i} \tag{3.3b}$$

$$m_i = 0.480 + 1.574 * \omega_i - 0.176 * \omega_i^2 \tag{3.3c}$$

$$\alpha_i = (1 + m_i * (1 - T_{Re,i}^{0.5}))^2 \tag{3.3d}$$

$$A_{p,i} = 0.42747 * \alpha_i * P_{Re,i}/T_{Re,i}^2 \tag{3.3e}$$

$$B_{p,i} = 0.08664 * P_{Re,i}/T_{Re,i} \tag{3.3f}$$

Binary A's:

$$A_{i,j} = (A_i * A_j)^{0.5} \tag{3.3g}$$

Mixture A and B:

$$A = \sum_{i}^{NC}\sum_{j}^{NC} x_i * x_j A_{i,j} * (1 - k_{i,j}) \tag{3.3h}$$

$$B = \sum_{i}^{NC} x_i * B_{p,i} \tag{3.3i}$$

For the compressibility of mixture:

$$Z^3 - Z^2 + Z(A - B - B^2) - AB = 0 \tag{3.3j}$$

For the fugacity of an i component:

$$\text{Correction for the fugacity: } C_i = \sum_{i}^{NC}\sum_{j}^{NC} x_j * A_{p,j}^{0.5} * (1 - K_{i,j}) \tag{3.3k}$$

$$ln\left(\frac{f_i}{P}\right) = (Z-1)\left(\frac{B_{p,i}}{B}\right) - ln(Z-B) - \frac{A}{B}\left(2C_i\frac{A_{p,i}^{0.5}}{A} - \frac{B_{p,i}}{B}\right) ln\left(\frac{Z+B}{Z}\right) \tag{3.3l}$$

The compressibility factor is obtained by solving equation 3.3j. This polynomial can have up to three roots. Several studies (Kamath et al., 2010; Gundersen, 1982; Zhao and Saha, 1998) have been carried out regarding the numerical implementation of this equation. One of the main questions corresponds whether to solve the equation independently from the rest of the system (For example: Flash calculations) or to include it, and if it is included how can it be included. On section 3.3.2, there is more information on how this equation is solved with the flash calculations.
Another challenge that arises and that has been pointed out by several studies is the selection of the roots of the equation. As shown by Kamath et al. (2010), the largest root is the compressibility for vapour phase and similarly the smallest root is for the liquid phase. In order to circumvent this challenge the algorithm used in this project requires a previous specification of the phase, so that no further criteria are required to choose the adequate Z.
Figure 3.2 shows the algorithm for solving the thermodynamics. This has been based on the work of Skogestad (2009). The code for this routine is based on Skogestad (2014)
Once the compressibility of the mixture, the values for the fugacity for the different components are calculated using equation 3.3l. This values are very important for the flash calculations and its use will be further explained on the corresponding section 3.3.2.

3.2.2 Enthalpy Calculations

The enthalpy for the mixture is calculated by adding the two following results:

- The ideal gas enthalpy of the mixture as the sum of the enthalpies of each component calculated by the integral of the heat capacity(Equation 3.4).

$$h_{ideal} = \int_{T_{ref}}^{T_f} C_p dT \tag{3.4}$$

- The departure enthalpy for the mix. This makes account for the non-idealities of the system and it is calculated based on the SRK model.

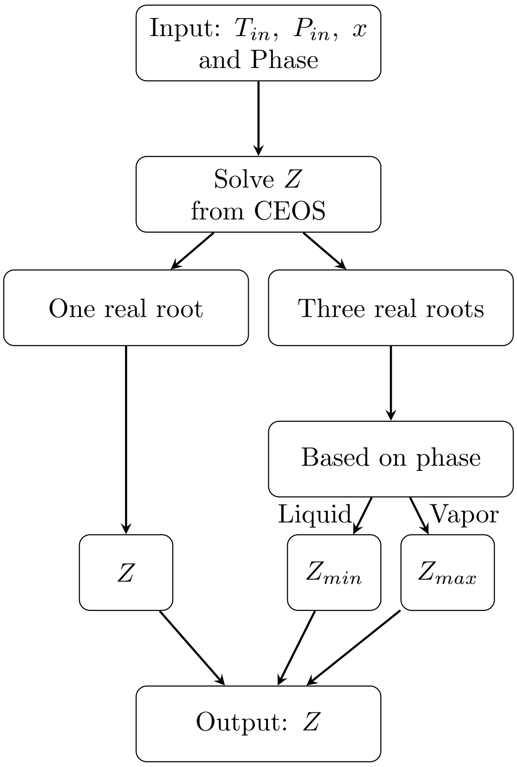


Figure 3.2: Algorithm for the calculation of Z

The ideal gas enthalpy is calculated for a reference temperature of 298.15 °C. The parameters for the equation are taken from Bruce, E Poling, John M. Prausnitz (1987).
Ideal gas enthalpy calculations:

$$h_{Ideal} = \sum_{i}^{NC} x_i h_{Ideal,i}$$

$$h_{Ideal,i} = A_i \left(T - T_{ref}\right) + \frac{1}{2} B_i \left(T^2 - T_{ref}^2\right) + \frac{1}{3} C_i \left(T^3 - T_{ref}^3\right) + \frac{1}{4} D_i \left(T^4 - T_{ref}^4\right) \tag{3.5}$$

The departure enthalpy is calculated to make up for all the non-idealities present in the system. This enthalpy is calculated using the compressibility determined by the cubic equation of state. The equations for the calcution of this departure property are based on Bruce, E Poling, John M. Prausnitz (1987) and Skogestad (2014). The steps for this calculation are registered on equation 3.6.

$$h_{departure} = \left(\frac{1}{\frac{BRT}{P}}\right) \left(\frac{AR^2T^2}{P} - T\frac{da}{dT}\right) log \left(\frac{Z}{Z+B} + RT(Z-1)\right) \tag{3.6a}$$

Where $\frac{da}{dT}$ for a multicomponent mixture is

$$\frac{da}{dT} = \sum_{i}^{NC}\sum_{j}^{NC} -\frac{R}{2}\sqrt{|\frac{0.42748}{T}|} x_i x_j \left(m_j \sqrt{|A_{p,i}\left(\frac{T_{c,j}}{P}\frac{T^2}{P_{c,j}R^2}\right)|} + m_i \sqrt{|A_{p,j}\left(\frac{T_{c,i}}{P}\frac{T^2}{P_{c,i}R^2}\right)|} \right) \tag{3.6b}$$

3.3 Unit Modelling

3.3.1 Model Specific Assumptions

Before further modelling it is necessary to make some additional assumptions. The first of these is to include both the MR Precooler and HA-01 into one unit. On practice the first unit (MR Precooler) brings the refrigerant to saturated vapor and then the other one (HA-01) partially condenses it. For the scope of this project it is assumed one big theoretical unit that brings the refrigerant to saturation and partially condenses it.

In this process regardless of the use of the degrees of freedom used, the different streams are assumed to be either vapor only, liquid only or vapour/liquid. This is to use more efficiently the flash calculations throughout the model and to minimize the possibility of convergence problems. Additionally, the pressure drops across all the heat exchangers are assumed to be negligible, thus the model has just two different pressures a high pressure (20 bar) and low pressure (3 bar). Lastly, the value for the split is assumed to be 50%. Figure 3.3 summarizes these assumptions.

Specific assumptions for the each unit will be explained in their corresponding sections. However, some remarks should be noted.

- The compressor uses a simplified model.
- Valves are isoenthalpic.
- For mixing, a PH flash calculation is carried out after a mass and energy balance at the mixing point.
- In order to account for the incoming energy from the environment there is an energy input on the stream connecting the unit HA-02 and the compressor.

Based on the last remark it is important to notice that the stream coming out from HA-02 is assumed to be two phases. In practice, this stream will be mostly vapour or have a very small liquid fraction.

Finally, Figure 3.4 contains the number of the streams and the nomenclature that will be used throughout the model.

3.3.2 Flash Calculations

There are several approaches to solve the flash calculations. For this case the calculations are based on the fugacities of the components and the mass balances. Equation 3.7 illustrates the relations used for the model.

$$K_i = \frac{\phi_{i,l}}{\phi_{i,v}} \tag{3.7a}$$

$$y_i = K_i x_i \tag{3.7b}$$

$$z_i = v_f y_i + (1 - v_f) x_i \tag{3.7c}$$

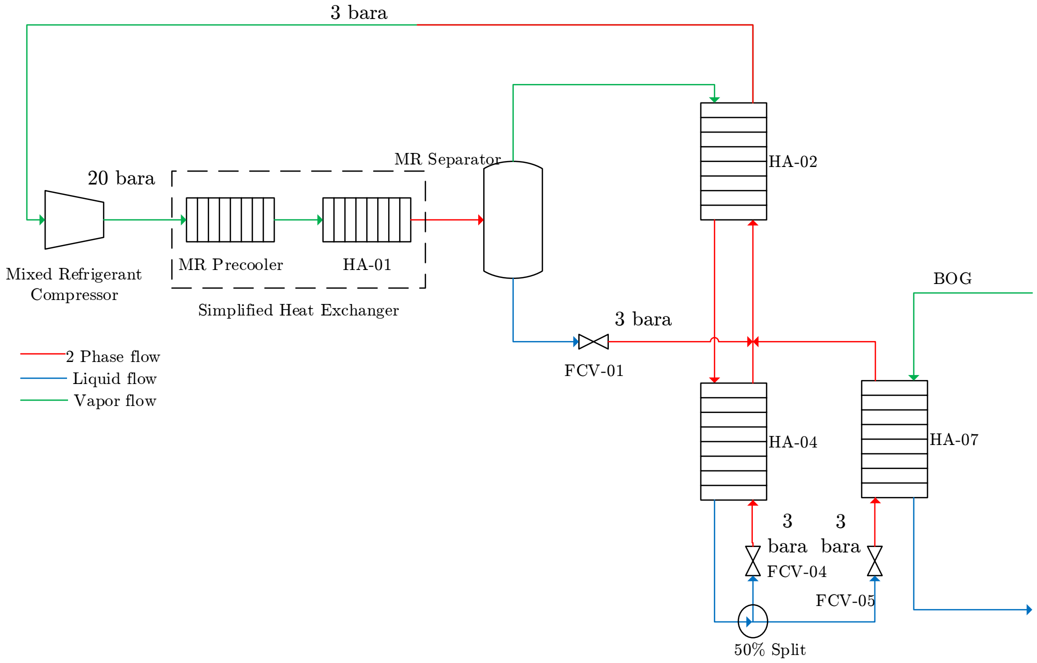


Figure 3.3: Main assumptions for the model

It should be noted that these equations use the fugacities, this so that the model takes into account the non-idealities of the system. The values for the fugacities are calculated from the cubic equation of state.

PT Flash Calculations

There are many ways to solve the cubic equation of state and the flash calculations. In this project the calculation is carried out using a simultaneous approach without including the compressibility as an additional state. Figure 3.5 illustrates the algorithm.
What this approach does is to internally solve the CEOS for the compressibility and feed that value to the flash calculations, which are solved at the same time. This method requires a good initial estimate for the vapour and liquid compositions as well as for the temperature and the vapour fraction. Additionally it should be noted that this is not the most robust method available. The robustness of the model can be improved including the compressibility as an additional state and simultaneously solving both the thermodynamics and the vapour liquid equilibrium. The code for this method is shown on Appendix B.2
These calculations are carried out in the following parts of the process:

- At the outlet of HA-01. A TP flash is used to calculate the conditions for stream 2, 7 well as for the outlets of the flash tank (streams 3 and 4)
- For all the sides of the heat exchangers in which the temperatures are known. These sides correspond to the streams 6 and 7.

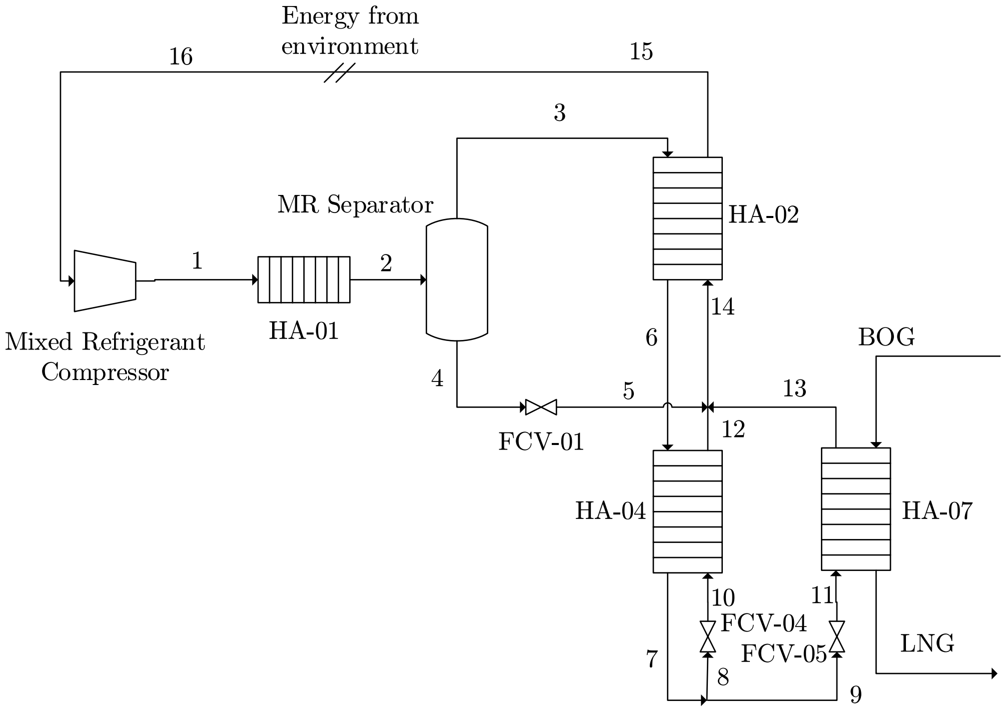


Figure 3.4: Nomenclature for the model

PH Flash Calculations

As for the case of the TP flash calculations, there are many ways to carry out a PH flash calculation. If there is an already working TP calculation it is possible to use an outer loop for the enthalpy, as described by Skogestad (2009). Considering how the TP flash has been implemented it is easier to include an additional condition corresponding to the energy balance (Equation 3.8) on the flash calculations. The code for this method is shown on Appendix B.3

$$h_f = v_f h_g + (1 - v_f)hl \tag{3.8}$$

Including these additions, figure 3.6 shows the algorithm for the calculation of the PH flash. This calculation is carried out several occasions:

- For the calculation of valves. As they are isoenthalpic and the output pressure is known, in order to find out the compositions of the different phases and its temperature it is necessarary to carry out a PH flash calculation.

- For the undefined side of the heat exchangers. The heat exchangers are calculated by finding out the duty of the side in which the output and input temperatures are known. With this duty, it is needed to calculate a PH flash to find out the output conditions of the undefined stream.

- For mixing. The mixing process that involves streams (5, 12 and 13) is carried out by

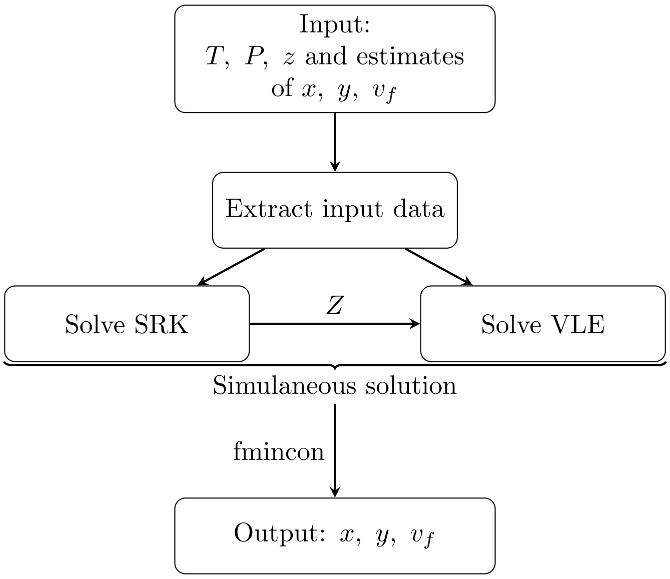


Figure 3.5: Algorithm for the TP flash calculation

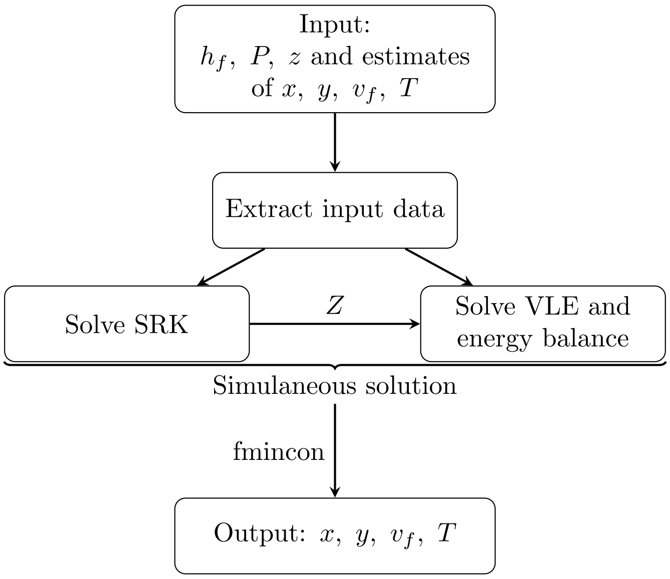

Input:
$h_f,\ P,\ z$ and estimates of $x,\ y,\ v_f,\ T$
Extract input data
Solve SRK
Z
Solve VLE and energy balance
Simulaneous solution
fmincon
Output: $x,\ y,\ v_f,\ T$

Figure 3.6: Algorithm for the PH flash calculation

calculating the mass and energy balances at that point. Once those are calculated, the conditions for stream 14 (the mixed stream) are calculated by a PH flash.

EO Flash Calculations

As mentioned on the previous sections, it is also possible to carry out the flash calculations in such a way that the value of the compressibility is explicit to the solver. This means that the implicit function for the calculation of the compressibility on the previous schemes is not used to calculate the value of Z, but instead to calculate the error between the Z as a state and the one that comes from the thermodynamic calculations.

$$Error_{liquid} = |SRK(T, P, z, liquid) - Z_l| \tag{3.9a}$$

$$Error_{vapour} = |SRK(T, P, z, vapour) - Z_v| \tag{3.9b}$$

Where $Error_{liquid}$ and $Error_{vapour}$ correspond to the error in the calculation of Z for liquid and vapour phase respectively. $SRK(T, P, z, phase)$(Appendix A.1) is the routine that calculates the roots of the CEOS based on algorithm 3.2 and Z_l and Z_v are the estimates of Z for each phase.
Using these additional states and constraints the full equations oriented algorithm for the flash calculation is shown on figure 3.7.

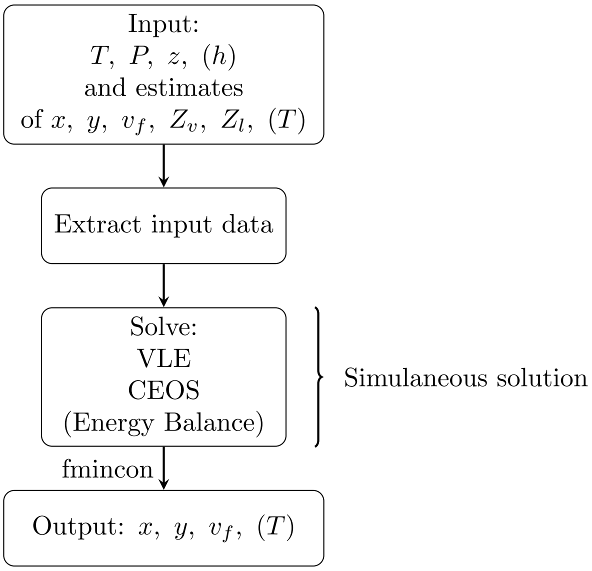


Figure 3.7: Algorithm for the EO flash calculations (PH flash in brackets)

This scheme was used as an initial approach towards a fully equations oriented model.The code for both TP and PH flash calculations is shown on Appendixes B.2 and B.3, respectively. On section 4.1, a comparison of the results using both approaches is shown.

Implementation

The implementation of the flash calculation was carried out based on the work of Lid and Skogestad (2008) and Skogestad (2009). This approach is based on an optimization problem in which the model is fully specified, thus the solution will work for any "Dummy" objective function.

In practice, this means that the flash calculations are implemented as nonlinear constrains and solved using the solver **fmincon** in Matlab.
A recursive routine has been set for the calculations, so that specific values such as temperature, pressure and enthalpy, are no longer fixed in the calculations but instead they can be modified.

3.3.3 Heat Exchangers

The modelling of the heat exchangers has been carried out in two general parts. One corresponds to the energy balances and the other to the sizing of the units. The energy balances are calculated in two different ways depending on the stream and the heat exchanger.

- Heat exchangers HA-02, HA-04 and HA-07. With streams 3, 6, 7, 11 and 13. For all of these streams the temperature is known, however the values for the enthalpy are unknown. In order to calculate the duties for these exchangers, the energy balance (equation 3.10) for the mentioned streams is carried out after carrying out a TP flash calculation for each stream.

$$Q_i = N_i(h_{out} - h_{in}) \tag{3.10}$$

- Heat exchangers HA-01, HA-02 and HA-04. With streams 1, 2, 10, 12, 14 and 15. In this case the inlet temperature (Streams 1, 10 and 14) and enthalpy is already known and so is the heat duty for the exchanger. In order to find out the output conditions from these units an energy balance is carried out (equation 3.11) and a PH flash calculation at the exit of the unit (Streams 2, 12 and 15).

$$N_i h_{in} + Q_i = N_i h_{out} \tag{3.11}$$

There is an special case, however for exchanger HA-07. This is the exchanger that liquefies the BOG. For this exchanger the duty is not know on neither side, however all sides have known temperatures but the flow is only known on the BOG side. The calculation was carried out by finding the duty of HA-07 using only the refrigerant streams assuming a molar flow of 1 Kmol/s. This duty was then compared to that of BOG and thus the refrigerant flow was adjusted so that both duties where equal.
The final step corresponds to the sizing of the units. This is carried out using equation 3.12 , these values might seem not relevant at this stage but these will provide operational constrains for the optimization problem.

$$Q = UALMTD \tag{3.12}$$

In this case as neither A nor U are known, the value UA will be the one used for the operational constrain on the optimization.

3.3.4 Compressor

As mentioned earlier the modelling of the compressor is simplified. It has been set for a recursive routine which follows the algorithm show on figure 3.8. The calculations for the compressor are described by equations 3.13 and 3.14.

$$\overline{C_p} = \sum_{i}^{NC} x_i C_{p,i} \tag{3.13a}$$

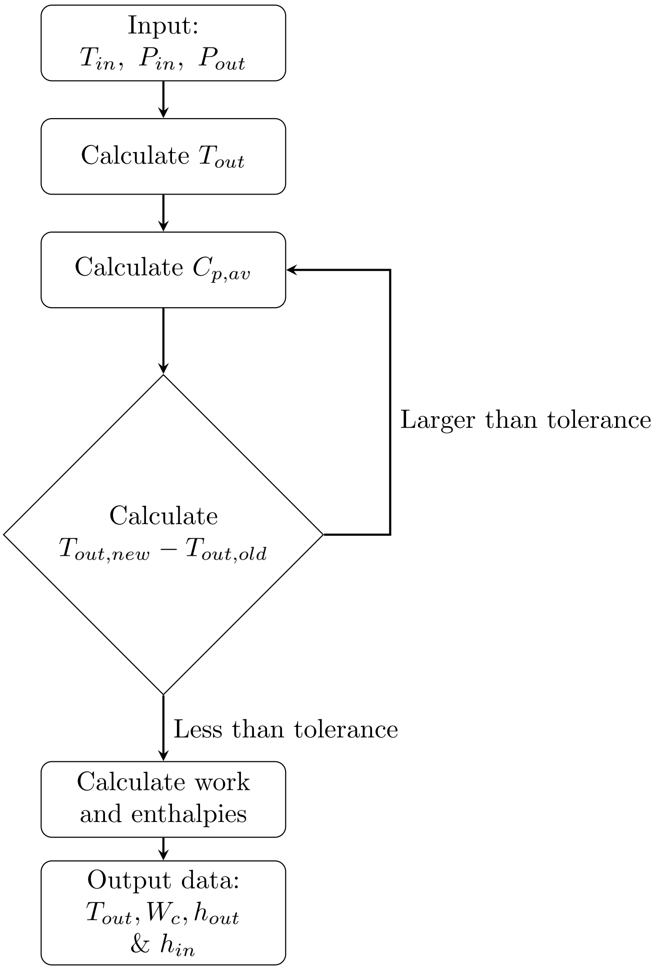


Figure 3.8: Compressor algorithm

where $\overline{C_p}$ is the weighted heat capacity and $C_{p,i}$ is defined as:

$$C_{p,i} = A_i + B_iT + C_iT^2 + D_iT^3 \tag{3.13b}$$

For the first iteration the outlet temperature is calculated based on the heat capacity at the inlet temperature$C_{p,in}$. The outlet temperature is calculated as follows:

$$T_{out} = T_{in} \left(\frac{P_{out}}{P_{in}}\right)^{\frac{R}{C_{p,in}}} \tag{3.13c}$$

Where R is the universal gas constant. Having a first estimate for the output temperature then:

$$T_{av} = \frac{T_{out} - T_{in}}{2} \tag{3.13d}$$

With this average temperature, it is possible to calculate the heat capacity for the mixture $C_{p,av}$. Then T_{out} is recalculated as follows:

$$T_{out} = T_{in} \left(\frac{P_{out}}{P_{in}} \right)^{\dfrac{R}{C_{p,av}}} \tag{3.13e}$$

$C_{p,av}$ can be recalculated and so can T_{out} until:

$$|T_{out,new} - T_{out,old}| < Tolerance \tag{3.13f}$$

The tolerance has been set to be 10^{-6}. Once the output temperature has been found the compressor work is calculated based on the inlet and outlet enthalpies.

$$W_C = N \frac{h_{in} - h_{out}}{\eta} \tag{3.14}$$

Where η is the compressor efficiency (set to 74%) and N the molar flow through the compressor.

Chapter 4

Results

4.1 Flash Calculations

As exposed earlier, two approaches for the flash calculations were considered. One with the compressibility factor as an explicit state and the other with this value as an implicit calculation. The calculations where performed using the same compositions and vapour fractions as estimates for a PT flash. The whole data for the initial points can be seen on Table 4.1. The results from this comparison are registered on Table 4.2.

Table 4.1: Initial point for the PT flash calculations using different approaches

Variable	Explicit Z	Implicit Z	Variable	Explicit Z	Implicit Z
x_{N_2}	0.0080	0.0080	y_{N_2}	0.1620	0.1620
x_{C1}	0.0860	0.0860	y_{C1}	0.4570	0.4570
x_{C2}	0.4590	0.4590	y_{C2}	0.3280	0.3280
x_{C3}	0.3450	0.3450	y_{C3}	0.0184	0.0184
x_{C4}	0.1020	0.1020	y_{C4}	0.0346	0.0346
Z_l	0.0751	-	Z_g	0.8923	-
$P[bar]$	20	20	v_f	0.65	0.65

Table 4.2: Results for the comparison between flash calculation approaches

Variable	Explicit Z	Implicit Z	Variable	Explicit Z	Implicit Z
x_{N_2}	0.0099	0.0099	y_{N_2}	0.1501	0.1501
x_{C1}	0.0875	0.0875	y_{C1}	0.4351	0.4351
x_{C2}	0.5212	0.5212	y_{C2}	0.3607	0.3607
x_{C3}	0.3232	0.3232	y_{C3}	0.0518	0.0518
x_{C4}	0.0583	0.0583	y_{C4}	0.0022	0.0022
Calculation Time$[s]$	0.037	0.035	v_f	0.6953	0.6953

4.2 Matlab Model

This section shows the results obtained for the model. There are some points that should be noted before exposing the results. Table 4.3 contains the information regarding the streams and table 4.4 has the information regarding the heat exchangers duty.

- Stream 15 can be considered as the tear stream of the problem, if a tear stream had to be defined.
- There was no solver for the tear stream, the convergence is achieved by calculating the total mass and energy balance before the calculation of stream 15.
- The simulation is carried out using Matlab. All the models are written anew except for the thermodynamics and flash calculations which are adapted from Skogestad (2014).

Table 4.3: Simulation results using Matlab

Stream	1	2	3	4	5	6	7	8
Vapour Fraction	1	0.6953	1	0	0.2707	0.4614	0	0
Temperature [K]	335.8	248.15	248.15	248.15	213.5	194.15	126.55	126.55
Pressure [bar]	20	20	20	20	3	20	20	20
Molar Flow [kgmole/s]	0.0434	0.0434	0.0302	0.0132	0.0132	0.0302	0.0302	0.0151
Molar Composition								
Nitrogen	0.1074	0.1074	0.1501	0.0099	0.0099	0.1501	0.1501	0.1501
Methane	0.3292	0.3292	0.4351	0.0875	0.0875	0.0875	0.4351	0.4351
Ethane	0.4096	0.4096	0.3607	0.5212	0.5212	0.3607	0.3607	0.3607
Propane	0.1345	0.1345	0.0518	0.3232	0.3232	0.0518	0.0518	0.0518
Butane	0.0193	0.0193	0.0022	0.0583	0.0583	0.0022	0.0022	0.0022

Stream	9	10	11	12	13	14	15	16
Vapour Fraction	0	0.1251	0.1251	0.9536	0.6402	0.6467	0.9351	1
Temperature [K]	126.55	115.2	115.2	200.1	171.65	194.7	219.9	240.15
Pressure [bar]	20	3	3	3	3	3	3	3
Molar Flow [kgmole/s]	0.0151	0.0151	0.0151	0.0151	0.0151	0.0434	0.0434	0.0434
Molar Composition								
Nitrogen	0.1501	0.1501	0.1501	0.1501	0.1501	0.1074	0.1074	0.1074
Methane	0.4351	0.4351	0.4351	0.4351	0.4351	0.3292	0.3292	0.3292
Ethane	0.3607	0.3607	0.3607	0.3607	0.3607	0.4096	0.4096	0.4096
Propane	0.0518	0.0518	0.0518	0.0518	0.0518	0.1345	0.1345	0.1345
Butane	0.0022	0.0022	0.0022	0.0022	0.0022	0.0193	0.0193	0.0193

Table 4.4: Heat duties for the heat exchangers using the Matlab model

Unit	Duty [kJ/s]
Compressor	232.7
HA_01	367.0
HA_02	246.4
HA_04	192.0
HA_07	106.0
Q_{Env}	88.9

4.3 Validation

In order to validate the model, a simulation using Aspen Hysys® has been carried out. In works such as Michelsen et al. (2010), Aspen Hysys® has been used as a validation tool for models built on Matlab.
For this model the following assumptions hold:

- The thermodynamics model used was SRK
- All of the assumptions for the units used in the Matlab model hold, such as the heat exchangers having no pressure drop.
- The needs for the BOG are assessed by Aspen Hysys®
- Based on the needs of the BOG, the refrigerant flow was set

Table 4.5 contains the stream data and table 4.6 contains the duties for the compressor and the heat exchangers.

Table 4.5: Simulation results using Aspen Hysys®

Stream	1	2	3	4	5	6	7	8
Vapour Fraction	1	0.7042	1	0	0.2731	0.4658	0	0
Temperature [K]	361.30	248.25	248.25	248.25	214.1	194.15	126.55	126.55
Pressure [bar]	20	20	20	20	3	20	20	20
Molar Flow [kgmole/s]	0.0434	0.0434	0.0306	0.0128	0.0128	0.0306	0.0306	0.0153
Molar Composition								
Nitrogen	0.1074	0.1074	0.1490	0.0084	0.0084	0.1490	0.1490	0.1490
Methane	0.3292	0.3292	0.4321	0.0841	0.0841	0.4321	0.4321	0.4321
Ethane	0.4096	0.4096	0.3637	0.5190	0.5190	0.3637	0.3637	0.3637
Propane	0.1345	0.1345	0.0529	0.3287	0.3287	0.0529	0.0529	0.0529
Butane	0.0193	0.0193	0.0023	0.0598	0.0598	0.0023	0.0023	0.0023

Stream	9	10	11	12	13	14	15	16
Vapour Fraction	0	0.1509	0.1509	0.9669	0.6377	0.6566	0.9470	1
Temperature [K]	126.55	115.20	115.20	201.91	171.70	191.50	221.10	221.10
Pressure [bar]	20	3	3	3	3	3	3	3
Molar Flow [kgmole/s]	0.0153	0.0153	0.0153	0.0153	0.0153	0.0434	0.0434	0.0434
Molar Composition								
Nitrogen	0.1490	0.1490	0.1490	0.1490	0.1490	0.1074	0.1074	0.1074
Methane	0.4321	0.4321	0.4321	0.4321	0.4321	0.3292	0.3292	0.3292
Ethane	0.3637	0.3637	0.3637	0.3637	0.3637	0.4096	0.4096	0.4096
Propane	0.0529	0.0529	0.0529	0.0529	0.0529	0.1345	0.1345	0.1345
Butane	0.0023	0.0023	0.0023	0.0023	0.0023	0.0193	0.0193	0.0193

Table 4.6: Comparison between the heat duties for the heat exchangers using the Matlab and Aspen Hysys® model

Unit	Hysys Duty [kJ/s]	Matlab	Error[%]
Compressor	255.9	232.7	9.1
HA_01	482.2	482.2	23.9
HA_02	254.6	246.4	3.2
HA_04	204.0	192.0	5.9
HA_07	110.0	106.0	3.6
Q_{Env}	62.3	88.9	42.7

Chapter 5

Discussion

The discussion for this project is carried out in three main areas, involving the main results obtained. As the scope of this project is to prepare the way for a detailed optimization study, the first section of the discussion will focus on the possible scenarios for the DOF analysis. The following two sections will consider the two criteria for the model (reliability and robustness).

5.1 Degrees of freedom

On section 3.1 an operations degrees of freedom was carried out. However, this analysis did not involve the kind of optimization to carry out (Skogestad, 2012). There exist two possible scenarios:

- Production optimization in which the aim is to used the system at its maximum possible capacity. In which case, the maximum flow can be a disturbance but the actual BOG flow is a DOF.
- Energy consumption optimization in which the aim is to minimize the energy consumption of the process for a given flow. In which case, The actual BOG flow is a disturbance.

Both of these cases have in common the degrees of freedom corresponding to the valves (FCV-04 and FCV -05), the compressor and the heat exchanger HA-01 (in practice it is the MR Precooler plus the unit HA-01). Nonetheless, depending on the case the flow of the BOG stream can become a disturbance.

For the energy consumption optimization, the flowrate and possible composition of the BOG stream will be among the most important perturbations. The process must ensure the condensation of the BOG for small disturbances on its flow using the minimum amount of energy and refrigerant. As for the maximum production, the BOG flow is going to maximized for a given flow of refrigerant.

Regardless of the kind of problem the number of DOFs will remain, but it is important to keep in mind which of the streams is the DOF for the chosen problem.

The definition of the optimization problem is out of the scope of this project. Nevertheless this degrees of freedom analysis sets up an initial framework for a detailed definition of the optimization problem.

5.2 Reliability

By comparing the results from Tables 4.3 and 4.5, it is possible to notice that the values of the simulation using Matlab are very close. The results for the vapour fraction, compositions and flows are very similar, which mean that the thermodynamics model has been successfully implemented in Matlab.
There is a difference on the temperature of stream 16 between the Aspen Hysys® and the Matlab model results. There is an increment of almost 20 K. This increment is due to a modification carried out on the Matlab model in order to ensure the existence of a vapour only stream at the inlet of the compressor. The reasoning for that modification is explained on section 5.3.
This difference can be noticed on the results for the duties of the heat exchanger HA-01 (Tables 4.4 and 4.6). However, despite that temperature difference the results(Table 4.6) for the other heat exchangers (HA-02,HA-04 and HA-07) are quite similar, with differences of less than 10% between the Matlab model and the Aspen Hysys® one.

5.3 Robustness

A challenge arose while carrying out the flash calculations. As stated earlier, the approach in which the flash calculations are carried out requires an initial estimate. This is a very important input for the calculations as a poorly initialization might lead to infeasible solutions.
In order to visualize the challenges for the flash calculations it is useful to have a look at the main refrigeration cycle (the one that liquefies the BOG, not the inner one) results (Figure 5.1). This plot includes the envelopes calculated on Aspen Hysys® for the two different refrigerant compositions being z_1 the initial refrigerant composition and z_2 the composition of the vapour stream after the phase separation. The number of the points on Figure 5.1 correspond to the number of the stream (3.4)

On Figure 5.1, it can be appreciated that the point 16 is very close to the envelope. The area around the envelope requires special attention. In this area is it possible for the solver to pass from a point which is in a two phase area to a single phase area. Once this situation happens, it is rarely possible to achieve convergence.
An attempt to circumvent this issue was to use the explicit compressibility approach over the implicit one. For calculations outside this stiff area close to the envelope, based on the results from Table 4.2, neither of the calculations offered a clear advantage. Additionally both approaches required special initialization or even modifications to converge for points close to the envelope. This convergence issue can lead to problems while carrying out the optimization algorithm, as the model will not converge. For this purpose the temperature of the stream 16 was modified by a trade off between robustness and reliability. By ensuring that the stream will be far enough from the envelope the robustness of the model is greatly improved, however this comes at the price of having results for the compressor and the heat exchanger HA-01 further from reality but still reliable.
Given the conditions of stream 16 and the amount of heat exchanged from the environment it is imperative to improve the flash calculation algorithms in such a way that these points which lie very close to the envelope can be calculated consistently.

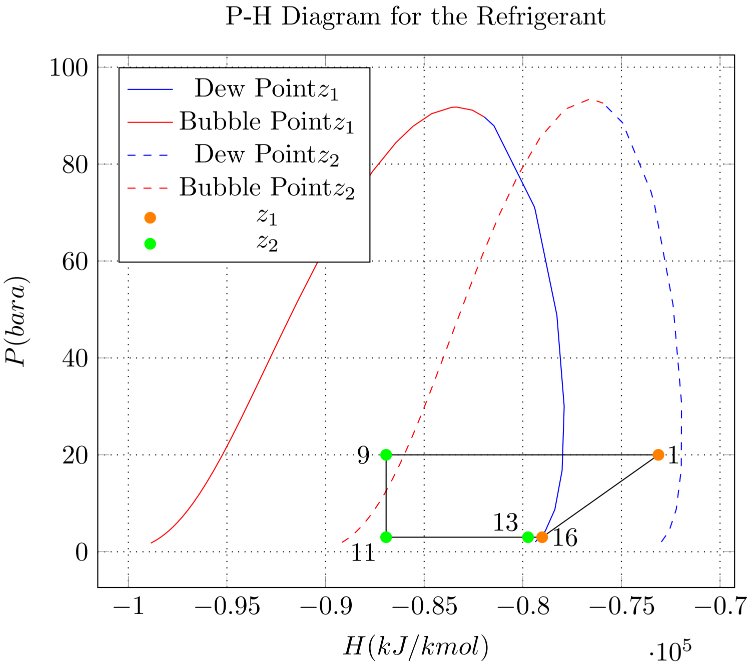


Figure 5.1: Main refrigeration cycle results

Chapter 6

Final Remarks

6.1 Conclusions

- In this project, it was possible to stablish an operation degrees of freedom analysis and further elaborate on it as a basis for a further optimization. This basis can be adapted depending on the objective function for the optimization problem to be defined.
- It was possible to generate a model which can describe the Mini-LNG plant for the conditions specified by Neksået al. (2010), despite some modifications in order to meet the robustness requirements.
- Neither of the flash calculations algorithms showed a considerable advantage for this model, in the analysed case. However, it is still possible to see that by using the algorithm, in which the compressibility is specified as an additional state, offers an advantage in cases where there are imaginary roots while solving the equation of state.
- In order to achieve a more reliable and robust model it is necessary to improve the flash calculation algorithms in such a way that the calculations for points close to the envelope do not required special attention.

6.2 Further work

The future work based on this project is given by the following points:

- Improve the robustness of the model towards optimization
  - Implement the model as a fully equations oriented model
  - Further improve the flash calculations for points close to the change phase boundaries
    - Improve the initialization for the algorithm
    - Modify the algorithm, so that convergence can be ensured for points close to the envelope.
- Further define the optimization problem
  - Maximum capacity or optimal energy consumption.
  - Values for the possible perturbations
  - Clearly define the constraints for the model
- Optimize the plant
  - Identify the conditions for which the constraints are active

Bibliography

Bjørn Austbø. *Use of Optimization in Evaluation and Design of Liquefaction Processes for Natural Gas.* PhD thesis, Norwegian University of Science and Technology, 2015. URL http://brage.bibsys.no/xmlui/bitstream/handle/11250/280210/NTNU{_}Bjorn{_}Austbo{_}PhD{_}avhandling{_}ny.pdf?sequence=1.

L.T. Biegler. Improved infeasible path optimization for sequential modular simulators—I: The interface. *Computers & Chemical Engineering*, 9(3):245–256, jan 1985. ISSN 00981354. doi: 10.1016/0098-1354(85)80003-X. URL http://www.sciencedirect.com/science/article/pii/009813548580003X.

L.T. Biegler and J.E. Cuthrell. Improved infeasible path optimization for sequential modular simulators—II: the optimization algorithm. *Computers & Chemical Engineering*, 9 (3):257–267, jan 1985. ISSN 00981354. doi: 10.1016/0098-1354(85)80004-1. URL http://www.sciencedirect.com/science/article/pii/0098135485800041.

John P. O'Connell Bruce, E Poling, John M. Prausnitz. *Properties of Gases and Liquids, Fifth Edition - Access Engineering from McGraw-Hill.* 1987. URL http://accessengineeringlibrary.com/browse/properties-of-gases-and-liquids-fifth-edition.

Truls Gundersen. Numerical aspects of the implementation of cubic equations of state in flash calculation routines. *Computers & Chemical Engineering*, 6(3):245–255, jan 1982. ISSN 00981354. doi: 10.1016/0098-1354(82)80015-X. URL http://www.sciencedirect.com/science/article/pii/009813548280015X.

Dag-Erik Helgestad. Modelling and optimization of the C3MR process for liquefaction of natural gas. Technical report, Norwegian University of Science and Technology (NTNU), 2010. URL http://www.nt.ntnu.no/users/skoge/diplom/prosjekt09/helgestad/Helgestad{_}project.pdf.

Magnus G. Jacobsen and Sigurd Skogestad. Optimization of LNG plants – challenges and strategies. *Computer Aided Chemical Engineering*, 29:1854–1858, 2011. ISSN 15707946. doi: 10.1016/B978-0-444-54298-4.50149-5. URL http://www.sciencedirect.com/science/article/pii/B9780444542984501495.

Jørgen Bauck Jensen and Sigurd Skogestad. Steady-State Operational Degrees of Freedom with Application to Refrigeration Cycles. *Industrial & Engineering Chemistry Research*, 48(14): 6652–6659, jul 2009. ISSN 0888-5885. doi: 10.1021/ie800565z. URL http://dx.doi.org/10.1021/ie800565z.

Ravindra S. Kamath, Lorenz T. Biegler, and Ignacio E. Grossmann. An equation-oriented approach for handling thermodynamics based on cubic equation of state in process optimization. *Computers & Chemical Engineering*, 34(12):2085–2096, dec 2010. ISSN 00981354. doi: 10.1016/j.compchemeng.2010.07.028. URL http://www.sciencedirect.com/science/article/pii/S0098135410002760.

Tore Lid and Sigurd Skogestad. Scaled steady state models for effective on-line applications. *Computers & Chemical Engineering*, 32(4-5):990–999, apr 2008. ISSN 00981354. doi: 10.1016/j.compchemeng.2007.04.003. URL http://www.sciencedirect.com/science/article/pii/S009813540700097X.

Finn Are Michelsen, Ivar Johan Halvorsen, Berit Floor Lund, and Per Eilif Wahl. Modeling and Simulation for Control of the TEALARC Liquified Natural Gas Process. *Industrial & Engineering Chemistry Research*, 49(16):7389–7397, aug 2010. ISSN 0888-5885. doi: 10.1021/ie901650e. URL http://dx.doi.org/10.1021/ie901650e.

W. Morton. Equation-Oriented Simulation and Optimization. *Proc Indian Natn Sci Acad*, 69(A):317–357, 2003. URL http://www.dli.gov.in/data{_}copy/upload/INSA/INSA{_}1/2000c4d6{_}317.pdf.

P. Nekså, E. Brendeng, M. Drescher, and B. Norberg. Development and analysis of a natural gas reliquefaction plant for small gas carriers. *Journal of Natural Gas Science and Engineering*, 2(2-3):143–149, July 2010. ISSN 18755100. doi: 10.1016/j.jngse.2010.05.001. URL http://www.sciencedirect.com/science/article/pii/S1875510010000326.

Mahdi Sharifzadeh and Nina F. Thornhill. Energy optimization for liquefaction process of natural gas in peak shaving plant. *Computer Aided Chemical Engineering*, 29:597–601, 2011. ISSN 15707946. doi: 10.1016/B978-0-444-53711-9.50120-6. URL http://www.sciencedirect.com/science/article/pii/B9780444537119501206.

Sigurd Skogestad. *Chemical and Energy Process Engineering.* CRC Press (Taylor & Francis Group), 2009.

Sigurd Skogestad. *Economic Plantwide Control.* John Wiley & Sons, Ltd, 2012. ISBN 9781119968962. doi: 10.1002/9781119968962.ch11. URL http://dx.doi.org/10.1002/9781119968962.ch11.

Sigurd Skogestad. Srk - flash. "http://www.nt.ntnu.no/users/skoge/book-cep/matlab/srk-flash/", 2014. URL http://www.nt.ntnu.no/users/skoge/book-cep/matlab/srk-flash/.

Giorgio Soave. Equilibrium constants from a modified Redlich-Kwong equation of state. *Chemical Engineering Science*, 27(6):1197–1203, jun 1972. ISSN 00092509. doi: 10.1016/0009-2509(72)80096-4. URL http://www.sciencedirect.com/science/article/pii/0009250972800964.

Ensheng Zhao and Sanjoy Saha. Applications of Complex Domain in Vapor-Liquid Equilibrium Calculations Using a Cubic Equation of State. *Industrial & Engineering Chemistry Research*, 37(5):1625–1633, may 1998. ISSN 0888-5885. doi: 10.1021/ie970639k. URL http://dx.doi.org/10.1021/ie970639k.

Appendices

Appendix A

Thermodynamics

Constants

The constants used for the calculations are registered in the following tables.

Table A.1: Critical properties and acentric factors for the componentsBruce, E Poling, John M. Prausnitz (1987)

Component	Nitrogen	Methane	Ethane	Propane	n-Butane
Critical pressure ($\times 10^5$) $[N/m^2]$	33.5	45.4	48.2	41.9	37.5
Critical temperature $[K]$	126.2	190.6	305.4	369.8	425.2
Acentric factor	0.040	0.008	0.098	0.152	0.193

Table A.2: Coefficients for the $C_p[kcal]$ calculation for the componentsBruce, E Poling, John M. Prausnitz (1987)

Coefficient	A	B	C	D
Nitrogen	7.440	-0.324×10^{-2}	6.400×10^{-6}	-2.790×10^{-9}
Methane	4.598	1.245×10^{-2}	2.86×10^{-6}	-2.703×10^{-9}
Ethane	1.292	4.254×10^{-2}	-1.657×10^{-5}	2.081×10^{-9}
Propane	-1.009	7.315×10^{-2}	-3.789×10^{-5}	7.678×10^{-9}
n-Butane	2.266	7.913×10^{-2}	-2.647×10^{-5}	-0.674×10^{-9}

A.1 SRK Calculations

This section includes the code for the SRK calculations.

```
%% SRK Package
function [Z,phi,H,V]= srks(x,T,P,phase)
% In: compositon x (vector), Temperature T [K], pressure p [N/m2], phase ...
    ['liquid' or 'vapor']
% OUT: Compressibility Z, Fugacity coeff.(vector) phi [-], enthalpy H [J/mol], ...
    Molar Volume V [m3/mol]
```

```

%Based on Linhart & Skogestad, 2008

%Components N2 C1 C2 C3 C4

NC=5;  % Number of components (1)Nitrogen - (2)Methane - (3)Ethane - (4)Propane ...
    - (5)Butane

%Component data from: Reid RC, Prausnitz JM, Poling BE. The properties of gases ...
    \& liquids (4th edition). New York: McGraw-Hill, Inc., 1987.
compData.Pc=[33.5 45.4 48.2 41.9 37.5 ]*1.e5;    % [N/m2]
compData.Tc=[126.2 190.6 305.4 369.8 425.2 ];  % [K]
compData.w=[0.040 0.008 0.098 0.152 0.193 ];   % acentric factor [-]
% Ideal has heat capacity (See Hid below):
compData.Cp(:, 1) = [7.440 -0.324e-2 6.400e-6 -2.790e-9]'; %Nitrogen
compData.Cp(:, 2) = [4.598 1.245e-2 2.86e-6 -2.703e-9]'; % Methane
compData.Cp(:, 3) = [1.292 4.254e-2 -1.657e-5 2.081e-9]'; % Ethane
compData.Cp(:, 4) = [-1.009 7.315e-2 -3.789e-5 7.678e-9]';    % Propane
compData.Cp(:, 5) = [2.266 7.913e-2 -2.647e-5 -0.674e-9]';    % n-butane
compData.Tref = 298.15;  % for Ideal gas heat capacity

kinteraction=zeros(NC,NC);      % Here: SRK binary interaction parameters set ...
    to zero

% initialize
liquid=1;vapor=2;
R = 8.314; %kJ/kmolK

Pc=compData.Pc;
Tc=compData.Tc;
w=compData.w;
ZRA=0.29056-0.08775*w;
Cp=compData.Cp;
Tref=compData.Tref;

% Calculations for given T, P  and composition (x)

%All equations from Soave,1972
Tre=T./Tc;
Pre=P./Pc;
m=0.480+1.574.*w-0.176.*w.^2; %Eq 15
a=(1+m.*(1-Tre.^0.5)).^2; %Eq 13
Ap=0.42747.*a.*Pre./Tre.^2; %Eq 8
Bp=0.08664.*Pre./Tre; % Eq 9

% Start calculations
% Binary a's:
Ab=zeros(NC,NC);
for i=1:NC
    for j=1:NC
        Ab(i,j)=(Ap(i)*Ap(j))^0.5;
    end
end
% Mixture a and b
A=0;
for i=1:NC
    for j=1:NC
        A=A+x(i)*x(j)*Ab(i,j)*(1-kinteraction(i,j));
    end
end
B=0;
for i=1:NC
```

```
    B=B+x(i)*Bp(i);
end

% Solve cubic equation to find compressibility Z = P*V/(R*T)
Zall=roots([1 -1 A-B-B^2 -A*B]);  %Eq 4 Soave,1972
% use real roots only
        Zreal=[];
    for i=1:3
      if isreal(Zall(i))==1
        Zreal=[Zreal Zall(i)];
      end
    end
% Select right root depending on phase
if      phase==liquid  Z=min(Zreal);
elseif phase==vapor   Z=max(Zreal);
else   write('error in specifying phase')
end

% Fugacity coefficient
% Correction added Nov. 2013:
for i = 1:NC
   corrphi(i) = 0;
for j = 1:NC
   corrphi(i) = corrphi(i) + x(j) * Ap(j)^0.5 * (1-kinteraction(i,j));
end
end
phi=exp((Z-1).*Bp/B-log(Z-B)...
     -A/B*log((Z+B)/Z).*((corrphi.*2.*Ap.^0.5./A)-Bp./B)); %Eq 21 Soave,1972

% Enthalpy Molar
dadT=0;
for i = 1:NC
    for j = 1:NC
        dadT = dadT -R / 2 * sqrt(abs(0.42748 / T)) * x(i) * x(j) * (m(j) * ...
            sqrt(abs(Ap(i)*(Tc(j)/P*(T^2)/(Pc(j))*(R^2)))) + m(i) * ...
            sqrt(abs(Ap(j) * (Tc(i)/P*(T^2)/(Pc(i))*(R^2))))) ;
    end
end

Hsrk=(1/ (B*R*T/P)) * ( A*R^2*T^2/P - T*dadT) * log(Z / (Z + B)) + R * T * (Z - ...
    1) ;

Hid=0;
for i = 1:NC
    Hid = Hid+(x(i)* (Cp(1,i) * (T - Tref) + 1/2 * Cp(2,i) * (T^2 - Tref^2) + ...
        1/3 * Cp(3,i) * (T^3 - Tref^3) + 1/4 * Cp(4,i) * (T^4 - Tref^4)));
end
%Conversion
Hid=Hid*4.184; %From kcal to kJ
H=Hid+Hsrk;
Hsrk;
Hid;
% Density (more precisely: molar volume)

if phase==liquid % Correct liquid SRK-volume using Peneleoux correction
     c=0;
     for i=1:NC
     c=c+x(i) * (0.40768 * (0.29441 - ZRA(i)) * (R * Tc(i)) / (Pc(i))) ;
     end
   V = ((Z * R * T / P)- c);
else  % vapor
   V = Z * R * T / P;
```

```
end
```

Appendix B

Individual Units Code

B.1 BOG Assessment

```

% % Assesessment of BOG Energy requirements

% This file calculates the energy required to be removed from the BOG gas
% in the system. This energy requirement Q_BOG, is the required duty for
% QHA_07 in the main code.

m=605; %Gas mass flow [Kg/h] (Neskaa, 2010)
z=[0.11 0.89 0 0 0]; %Gas Mole compositions
M_N= 28; %Nitrogen (N2) molecular mass [Kg/Kmol]
M_C1=16; %Methane (CH4) molecular mass [Kg/Kmol]
n= 605/(z(1)*M_N+z(2)*M_C1)*(1/3600); %BOG Molar Flow [Kmol/s]

Tin = -32 + 273.15; %Inlet temperature [K]
Tout = -154 +273.15; %Outlet temperature [K]
P = 18.2e5;% Pressure [Pa] (Assumed constant through the exchanger)

[Zin,phiin,Hin,Vin]= srks(z,Tin,P,2); %Inlet SRK Calculations. Initial phase ... vapor only.
[Zout,phiout,Hout,Vout]= srks(z,Tout,P,1); %Outlet SRK Calculations. Outlet ... liquid only.

Q_BOG= n*(Hout-Hin); %Heat flow from the gas [Kj/s]

QHA_07= 7.3768e+03; %Heat flow [Kj/s] for refrigerant mole flow = 1 Kmol/s;

%This solves a simplified version of the main model to calculate the required flow
global Q
Q=Q_BOG;
w0=0.013;
lb=0;
ub=Inf;
N_ref=fmincon('1',w0,[],[],[],[],lb,ub,@nfunc)

% Save information to file
filename='BOGresults.mat';
save(filename,'N_ref','Q_BOG')
```

B.2 TP Flash

This section counts for two parts. The first is the flash equations in the way of nonlinear constrains for the solver. The second corresponds to the recursive routine for he solver to use different conditions for the calculations.

Base Routine

```

function [c,ceq]=f(w)
% This is file flashPTsrk.m
%    Flash at given pressure and temperature
%    w = internal state vector of length 2*NC-1
% Usage for a case with NC=5 components (note: give compositions only for NC-1 ...
    components):
%    x0=Initial estimate x, y0= initial estimate y; VF0= initial estimate vf;  ...
    w0= [x0, y0, VF0];
%    w=fmincon('1',w0,[],[],[],[],[],[],@flashPTsrk);
%    NC=(length(w)+1)/2; x =  [w(1:NC-1) 1-sum(w(1:NC-1))]; y =[w(NC:2*NC-2) ...
    1-sum(w(NC:2*NC-2))]; VF=w(2*NC-1);
%   Vapor_fraction=VF, Liquid_Composition = x, Vapor_Composition = y
% Definition of global variables for recursive routine
global T p zf
% All in SI units
% T = Temperature [K]
% p = Pressure [N/m^2]
% zf = Initial composition

%Number of components
NC=5;

liquid=1;
vapor=2;
% Extract states:
x(1:NC-1)=w(1:NC-1);                   % liquid composition
y(1:NC-1)=w(NC:2*NC-2);                % vapor composition
VF =w(2*NC-1);                         % vapor fraction
% Composition of last component:
x(NC)=1-sum(x(1:NC-1));
y(NC)=1-sum(y(1:NC-1));

% Fugacity, enthalpy and volumes from SRK
[Zl,phil,hl,Vlm]=srks(x,T,p,liquid);       % liquid
[Zg,phig,hg,Vgm]=srks(y,T,p,vapor);        % vapor

K = phil./phig;

f1 = y - K.*x;                                                   % = 0   Algebraic ...
     (NC): VLE
f2(1:NC-1) = zf(1:NC-1) - VF.*y(1:NC-1) - (1-VF).*x(1:NC-1);     % = 0   ...
    Algebraic(NC-1): Component mass balances

c=[]; ceq=[f1';f2'];
```

Recursive Routine

```
function [ x, y,VF, hl, hv  ] = flashTPeo(t,P, x0,y0,z,VF0,Zl0,Zg0)
% % Recursive routine for TPFlash calculation.
% Extracts liquid composition, vapor composition, vapour
% fraction, molar enthalpy for liquid and for vapor phase, given Temperature ...
    and Pressure.
%Definition of global variables for recoursive routine
global T p zf
T=t; p=P; zf=z;
% zf % Specified feed composition
% p % Specified pressure [N/m2]
% T % Specified temperature [K]

%Inital estimates for solver
w0= [x0, y0, VF0,Zl0,Zg0];
%x0 initial estimate for liquid composition
%y0 initial estimate for vapor composition
%vf0 initial estimate for vapor fraction
%Zl0 initial estimate for liquid compressibility
%Zg0 initial estimate for vapor compressibility
NC=5;

%Boundary constrains.
UB(1:2*NC+1)=1; UB(2*NC+1)=3;
LB(1:2*NC+1)=0;

%Solution subject to the equilibrium and solving the EOS. For further
%details check flashTPsrk.m
w=fmincon('1',w0,[],[],[],[],LB,UB,@flashTPsrkeo);

% %Data extraction

%Compositions for each phase (Final component calculated)
x =[w(1:NC-1) 1-sum(w(1:NC-1))];
y =[w(NC:2*NC-2) 1-sum(w(NC:2*NC-2))];
VF=w(2*NC-1);

%Compresibility, fugacity, enthalpy and volume calculations
%For the purpose of this project, only enthalpy is needed.
[Zv,phiv,hv,Vv]= srks(y,T,p,2);
[Zl,phil,hl,Vl]= srks(x,T,p,1);

end
```

Base Routine EO

```

function [c,ceq]=f(w)
% This is file flashTPsrkeo.m
%    Flash at given pressure and temperature with explicit fugacities
%    w = internal state vector of length 2*NC-1
% Usage for a case with NC=5 components (note: give compositions only for NC-1 ...
    components):
%   x0=Initial estimate x, y0= initial estimate y; VF0= initial estimate vf; ...
    Zl0=initial estimate kiquid compressibility;
%   Zg0= Initial estimate vapor compressibility; w0= [x0, y0, VF0,Zl0,Zg0];
% Definition of global variables for recursive routine
global T p zf
% All in SI units
% T = Temperature [K]
% p = Pressure [N/m^2]
% zf = Initial composition

%Number of components
NC=5;



% Extract states:
x(1:NC-1)=w(1:NC-1);                    % liquid composition
y(1:NC-1)=w(NC:2*NC-2);                 % vapor composition
VF =w(2*NC-1);                          % vapor fraction
Zl =w(2*NC);                            % Liquid compressibility
Zg = w(2*NC+1);                         % Vapor compressibility

% Composition of last component:
x(NC)=1-sum(x(1:NC-1));
y(NC)=1-sum(y(1:NC-1));

% Fugacity, enthalpy and volumes from SRK
[phil,hl,Vlm,errl]=srkseo(x,T,p,Zl);        % liquid
[phig,hg,Vgm,errg]=srkseo(y,T,p,Zg);         % vapor

K = phil./phig;


f1 = y - K.*x;                                              % = 0   Algebraic ...
    (NC): VLE
f2(1:NC-1) = zf(1:NC-1) - VF.*y(1:NC-1) - (1-VF).*x(1:NC-1);  % = 0   ...
    Algebraic(NC-1): Component mass balances
f3 = [abs(errl) abs(errg)];
c=[]; ceq=[f1';f2';f3'];
```

Recursive Routine EO

```
function [ x, y,VF, hl, hv  ] = flashTP(t,P, x0,y0,z,VF0)
% % Recursive routine for TPFlash calculation.
% Extracts liquid composition, vapor composition, vapour
% fraction, molar enthalpy for liquid and for vapor phase, given Temperature ...
    and Pressure.
%Definition of global variables for recoursive routine
global T p zf
T=t; p=P; zf=z;
% zf % Specified feed composition
% p % Specified pressure [N/m2]
% T % Specified temperature [K]

%Inital estimates for solver
w0= [x0, y0, VF0];
%x0 initial estimate for liquid composition
%y0 initial estimate for vapor composition
%vf0 initial estimate for vaporr fraction

%Boundary constrains. All the states must lie between 0 and 1.
lb=zeros(size(w0));
ub=ones(size(w0));

```

```
%Solution subject to the equilibrium and solving the EOS. For further
%details check flashTPsrk.m
w=fmincon('1',w0,[],[],[],[],lb,ub,@flashTPsrk);

% %Data extraction
NC=(length(w)+1)/2;
%Compositions for each phase (Final component calculated)
x =[w(1:NC-1) 1-sum(w(1:NC-1))];
y =[w(NC:2*NC-2) 1-sum(w(NC:2*NC-2))];
VF=w(2*NC-1);

%Compresibility, fugacity, enthalpy and volume calculations
%For the purpose of this project, only enthalpy is needed.
[Zv,phiv,hv,Vv]= srks(y,T,p,2);
[Zl,phil,hl,Vl]= srks(x,T,p,1);

end
```

B.3 PH Flash

This section counts for two parts. The first is the flash equations in the way of nonlinear constrains for the solver. The second corresponds to the recursive routine for he solver to use different conditions for the calculations.

Base Routine

```
function [c,ceq]=f(w)
% This is file flashPHsrk.m
%    Flash at given pressure and enthalpy
%    w = internal state vector of length 2*NC
% Usage for a case with NC=5 components (note: give composition only for NC-1 ...
    component):
%    x0= Initial estimates x, y0= Initial estimates yo; VF0= Initial estimate ...
    Vf; T0= Initial estimate T;
%    Temperature=T, Vapor_fraction=VF, Liquid_Composition = x, ...
    Vapor_Composition = y
%Global variables for recursive routine
global hf p zf

% % Data (parameters and independent variables)
%Number of components
NC=5;
% zf % Specified feed composition
% p % Specified pressure [N/m2]
% hf % Specified enthalpy [kJ/Kmol] (ref: h=0 for ideal gas at 298K)

liquid=1;vapor=2;
R=8.13;                                 % kJ/Kmol K

% Extract states:
x(1:NC-1)=w(1:NC-1);                    % liquid composition
y(1:NC-1)=w(NC:2*NC-2);                 % vapor composition
VF = w(2*NC-1);                         % Vapor fraction
T =w(2*NC);                             % Temperature [K]

% Composition of last component:
```

```
x(NC)=1-sum(x(1:NC-1));
y(NC)=1-sum(y(1:NC-1));

% Fugacity, enthalpy and volumes from SRK
[Zl,phil,hl,Vlm]=srks(x,T,p,liquid);        % liquid
[Zg,phig,hg,Vgm]=srks(y,T,p,vapor);         % vapor

K = phil./phig;

f1 = y - K.*x;                                        % = 0   Algebraic ...
    (NC): VLE
f2(1:NC-1) = zf(1:NC-1) - VF.*y(1:NC-1) - (1-VF).*x(1:NC-1);  % = 0   ...
    Algebraic(NC-1): Component mass balances
f3 = hf - VF*hg - (1-VF)*hl;                          % = 0   Algebraic ...
    (1): Energy balance

c=[]; ceq=[f1';f2';f3'];
```

Recursive Routine

```
function [x, y, VF, T  ] = flashPH( P, h,  x0, y0, z, T0, VF0 )
% %Recursive function for the claculation of a PH flash.
% Extracts liquid composition, vapor composition, vapour
% fraction and temperature for liquid and for vapor phase, given Pressure and ...
    enthalpy.
% Definition of global variables for recursive routine.
global p zf hf
p=P; hf=h; zf=z;
% zf % Specified feed composition
% p % Specified pressure [N/m2]
% hf % Specified enthalpy [kJ/Kmol] (ref: h=0 for ideal gas at 298K)

% Initialization vector
w0= [x0, y0, VF0, T0];
%x0 initial estimate for liquid composition
%y0 initial estimate for vapor composition
%vf0 initial estimate for vaporr fraction
%T0 intial estimate for temperature

%Boundary constrains. All the states except temperature must lie between 0 and 1.
lb=zeros(size(w0));
ub=ones(size(w0));
ub(10)=Inf;
%Inequality constrains. The added compositions cannot be larger than 1.
A=[1 1 1 1 0 0 0 0 0 0;
    0 0 0 0 1 1 1 1 0 0];
B=[1; 1];
%Solver options
options = optimoptions('fmincon','Algorithm','interior-point','TolCon', ...
    1e-6,'TolFun', 1e-4, 'MaxIter', 3000, 'TolX', 1e-12, ...
    'ScaleProblem','obj-and-constr');

%Solution subject to the equilibrium and solving the EOS. For further
%details check flashPHsrk.m
[w,fval,exitflag,output]=fmincon('1',w0,A,B,[],[],lb,ub,@flashPHsrk,options);

% %Data extraction
NC=(length(w)+1)/2;
```

```

%Liquid and vapor compositions
x =  [w(1:4) 1-sum(w(1:4))];
y =[w(5:8) 1-sum(w(5:8))];
%Vapor fraction
VF=w(9);
%Temperature
T = w(10);
end
```

Base Routine EO

```
function [c,ceq]=f(w)
% This is file flashPHsrkeo.m
%    Flash at given pressure and enthalpy
%    w = internal state vector of length 2*NC
% Usage for a case with NC=5 components (note: give composition only for NC-1 ... component):
%    x0= Initial estimates x, y0= Initial estimates yo; VF0= Initial estimate ... Vf; T0= Initial estimate T;
%    Zl0=initial estimate kiquid compressibility; Zg0= Initial estimate vapor ... compressibility; w0= [x0, y0, VF0, T0, Zl0, Zg0];
%    Temperature=T, Vapor_fraction=VF, Liquid_Composition = x, ... Vapor_Composition = y
%Global variables for recursive routine
global hf p zf

% % Data (parameters and independent variables)
%Number of components
NC=5;
% zf % Specified feed composition
% p % Specified pressure [N/m2]
% hf % Specified enthalpy [kJ/Kmol] (ref: h=0 for ideal gas at 298K)


% Extract states:
x(1:NC-1)=w(1:NC-1);                   % liquid composition
y(1:NC-1)=w(NC:2*NC-2);                % vapor composition
VF = w(2*NC-1);                        % Vapor fraction
T =w(2*NC);                            % Temperature [K]
Zl = w(2*NC+1);                        % Liquid Compressibility
Zg = w(2*NC+2);                        % Gas Compressibility
% Composition of last component:
x(NC)=1-sum(x(1:NC-1));
y(NC)=1-sum(y(1:NC-1));

% Fugacity, enthalpy and volumes from SRK
[phil,hl,Vlm,errl]=srkseo(x,T,p,Zl);       % liquid
[phig,hg,Vgm,errg]=srkseo(y,T,p,Zg);        % vapor


K = phil./phig;

f1 = y - K.*x;                                              % = 0   Algebraic ... (NC): VLE
f2(1:NC-1) = zf(1:NC-1) - VF.*y(1:NC-1) - (1-VF).*x(1:NC-1);  % = 0   ... Algebraic(NC-1): Component mass balances
```

```
f3 = hf - VF*hg - (1-VF)*hl;                                    % = 0   Algebraic ...
    (1): Energy balance
f4 = [abs(errl) abs(errg)];

c=[]; ceq=[f1';f2';f3';f4'];
```

Recursive Routine EO

```
function [x, y, VF, T  ] = flashPHeo( P, h,  x0, y0, z, T0, VF0, Zl0, Zg0 )
% %Recursive function for the claculation of a PH flash.
% Extracts liquid composition, vapor composition, vapour
% fraction and temperature for liquid and for vapor phase, given Pressure and ...
    enthalpy.
% Definition of global variables for recursive routine.
global p zf hf
p=P; hf=h; zf=z;
% zf % Specified feed composition
% p % Specified pressure [N/m2]
% hf % Specified enthalpy [kJ/Kmol] (ref: h=0 for ideal gas at 298K)
NC=5;
% Initialization vector
w0= [x0, y0, VF0, T0, Zl0, Zg0];
%x0 initial estimate for liquid composition
%y0 initial estimate for vapor composition
%vf0 initial estimate for vaporr fraction
%T0 intial estimate for temperature

%Boundary constrains.
UB(1:2*NC-1)=1;UB(2*NC)=Inf; UB(2*NC+1:2*NC+2)=3;
LB(1:2*NC+2)=0;
%Solver options
options = optimoptions('fmincon','Algorithm','interior-point','TolCon', ...
    1e-6,'TolFun', 1e-4, 'MaxIter', 3000, 'TolX', 1e-12);

%Solution subject to the equilibrium and solving the EOS. For further
%details check flashPHsrk.m
[w,fval,exitflag,output]=fmincon('1',w0,[],[],[],[],LB,UB,@flashPHsrkeo,options);

%Liquid and vapor compositions
x =  [w(1:4) 1-sum(w(1:4))];
y =[w(5:8) 1-sum(w(5:8))];
%Vapor fraction
VF=w(9);
%Temperature
T = w(10);
end
```

Compressor

```
function [ Tout1 ,W,hout,hin] = compressor( Pin,Tin,Pout,x,N)
%This function corresponds to the recursive routine to calculate the
%compressor. This function uses a simplified isoenthropic model.
% % Input
% Pin = Inlet pressure [Pa]
% Tin = Inlet temperature [K]
% Pout = Outlet pressure [Pa]
% x = Inlet molar composition
% N = Inlet molar flow [Kmol/s]
% % Output
% Tout1 = Outlet temperature [K]
% W = Compressor work [Kj/s]
% hout = Outlet molar enthalpy [Kj/Kmol]
% hin = Inlet molar enthalpy [Kj/Kmol]

% % First estimation of the average Cp
c=cpav(Tin,x); % Calculates the average Cp [Kj/K] for a given mixture at a ...
    especified temperature.
R = 8.314; % Universal gas constant [Kj/KmolK]
Tout1=Tin*(Pout/Pin)^(R/c); %Initial calculation for the output temperature
b=1; % Initial value for difference
n=0.74; %Compressor efficiency

% % Loop to find a better estimate of the themperature.
while b > 1e-6 %Tolerance between calculations 1e-6
Ta=(Tin+Tout1)/2; % Average temperature
c=cpav(Ta,x); % Cp for average temperature
Tout=Tin*(Pout/Pin)^(R/c); % New estimate
b=abs(Tout1-Tout); % Difference between new and old
Tout1=Tout; % Overwrite last estimate
end

%Once the temperature is set, calculates enthalpy for in and outlet
%conditions.
[Zi,phii,Hi,Vi]= srks(x,Tin,Pin,2);
[Zo,phio,Ho,Vo]= srks(x,Tout1,Pout,2);

%Compressor work is calculated
W=N*((Ho-Hi)/n);

%Enthalpies extracted
hout=Ho;
hin=Hi;
end
```

Appendix C

Overall code

Main code

```
global T p zf hf
clear all
clc
% % Main Model code
% This code includes a steady state simulation of the whole mini LNG plant.

%Initialize variables
P=zeros(1,17); % Pressures [pa]
Ts=zeros(1,17); % Stream temperature [K]
n=ones(1,17); % Flows in [kmol/s]
h=zeros(1,17); % Molar enthalpy [Kj/kmol]
vf=zeros(1,17); % Vapor Fraction
H=zeros(1,17); % Total Enthalpy [Kj]
x=zeros(17,5); %Liquid Composition
y=zeros(17,5); %Vapor Composition
z=zeros(17,5); %Total Composition


%Set the pressures for the whole system
lp=[15 14 12 5 13 10 11 16 17];
hp=[1 2 3 4 6 7 8 9];
P(lp)=3e5;
P(hp)=20e5;

%Set known compositions

z(1,:)= [0.1074 0.3292 0.4096 0.1345 0.0193];
z(2,:)= [0.1074 0.3292 0.4096 0.1345 0.0193];
z(16,:)= [0.1074 0.3292 0.4096 0.1345 0.0193];

%Vapor
y(1,:)= [0.1074 0.3292 0.4096 0.1345 0.0193];
y(16,:)= [0.1074 0.3292 0.4096 0.1345 0.0193];

% Set known flows
%Load the results from BOG.m
filename='BOGresults.mat';
load(filename)
```

```
N = N_ref; %Refrigerant flow [Kmol/s]
%N=0.043; %Estimate
n=N*n;

%Set known Temperatures
Ts(16)=-33+273.15;%Modified to make it vapour only
Ts(2)=-25+273.15;
Ts(6)=273.15-79;
Ts(7)=273.15-146.6;
Ts(8)=Ts(7);
Ts(9)=Ts(7);
Ts(13)=273.15-101.5;

%Set known vapor fractions
vf(16)=1;
vf(1)=1;
vf(3)=1;
vf(4)=0;

%Compressor
[Ts(1),Wc,h(1),h(16)]= compressor( P(16),Ts(16),P(1),z(16,:),n(1));
H(1)=n(1)*h(1);
H(16)=n(16)*h(16);

%Flash calculation
x0=[0.0080 0.0860 0.4590 0.3450];
y0=[0.1620 0.4570 0.3280 0.1840];
VF0=0.65;
[ x(2,:), y(2,:),vf(2), h(4), h(3)] = flashTP(Ts(2),P(2), x0,y0,z(2,:),VF0);
h(2)=h(3)*vf(2)+h(4)*(1-vf(3));

%Assign flows and compositions
z(3,:)=y(2,:);
z(4,:)=x(2,:);
n(3)=n(2)*vf(2);
n(4)=n(2)*(1-vf(2));
H(3)=n(3)*h(3);
H(4)=n(4)*h(4);
Ts(3)=Ts(2);
Ts(4)=Ts(2);

%Heat exchanger HA-01
%By assuming adiabatic flash
H(2)=H(3)+H(4);
QHA_01=H(1)-H(2);

%Valve FCV-01
%Isoenthalpic Valve
h(5)=h(4);
n(5)=n(4);
z(5,:)=z(4,:);
x0=[0.1 0.1 0.413 0.437]; y0=[0.032 0.296 0.588 0.079]; VF0=0.26; T0=200;
[x(5,:), y(5,:), vf(5), Ts(5)] = flashPH( P(5), h(5), x0, y0, z(5,:), T0, VF0 );
H(5)=n(5)*h(5);


%Heat exchanger HA-02 first side
%Calculate stream 6
z(6,:)=z(3,:);
n(6)=n(3);
x0=[0.02 0.26 0.59 0.11];y0=[0.28 0.63 0.0 0.001]; VF0=0.5;
[ x(6,:), y(6,:),vf(6), hl, hv] = flashTP(Ts(6),P(6), x0,y0,z(6,:),VF0);
```

```
h(6)=hv*vf(6)+hl*(1-vf(6));
H(6)=h(6)*n(6);
QHA_02=H(3)-H(6);

%Calculate HA-04 first side
%Calculate stream 7
z(7,:)=z(6,:);
n(7)=n(6);
vf(7)=0; % Assumed liquid
[Zv,phiv,h(7),Vv]= srks(z(7,:),Ts(7),P(7),1);
H(7)=h(7)*n(7);
QHA_04=H(6)-H(7);

%Split
Sf=0.5;
z(8,:)=z(7,:);
z(9,:)=z(7,:);
vf(8)=vf(7);
vf(9)=vf(7);
n(8)=n(7)*Sf;
n(9)=n(7)*(1-Sf);
h(8)=h(7);
h(9)=h(7);
H(8)=h(8)*n(8);
H(9)=h(9)*n(9);
Ts(8)=Ts(7);
Ts(9)=Ts(7);

%FCV-04 // FCV-05
z(10,:)=z(8,:);
n(10)=n(8);
h(10)=h(8);
x0=[0.045 0.5 0.38 0.06]; y0=[0.75 0.24 0.01 0]; VF0=0.1; T0=100;
[x(10,:), y(10,:), vf(10), Ts(10)] = flashPH( P(10), h(10), x0, y0, z(10,:), ...
    T0, VF0 );
H(10)=n(10)*h(10);
%Stream 11
z(11,:)=z(9,:);
n(11)=n(9);
h(11)=h(9);
Ts(11)=Ts(10);
x(11,:)=x(10,:);
y(11,:)=y(10,:);
vf(11)=vf(10);
H(11)=n(11)*h(11);

%HA-04 Second side
n(12)=n(10);
H(12)=H(10)+QHA_04;
h(12)=H(12)/n(12);
z(12,:)=z(10,:);
x0=z(12,1:4); y0=x0; VF0=1; T0=200;
[x(12,:), y(12,:), vf(12), Ts(12)] = flashPH( P(12), h(12), x0, y0, z(12,:), ...
    T0, VF0 );


%HA-07 Simplified
%Global energy balance
%This energy is adjusted to the requirements of the natural gas by modifying ...
    the refrigerant flow.
%These requirements are in BOG.m.
z(13,:)=z(11,:);
```

```
n(13)=n(11);
x0=[0.0016 0.0786 0.7187 0.1879]; y0=[0.1420 0.7233 0.1327 0.0021];
VF0=0.7;
[ x(13,:), y(13,:),vf(13),hl, hv] = flashTP(Ts(13),P(13), x0,y0,z(13,:),VF0);
h(13)=vf(13)*hv+(1-vf(13))*hl;
H(13)=h(13)*n(13);
QHA_07=H(13)-H(11);

% %Mixer
n(14)=n(5)+n(12)+n(13);
z(14,:)= (n(5).*z(5,:)+n(12).*z(12,:)+n(13).*z(13,:))/n(14);
H(14)=H(5)+H(12)+H(13);
h(14)=H(14)/n(14);
x0=[0.0009 0.0360 0.4746 0.3975]; y0=[0.1094 0.5867 0.2820 0.0214]; VF0=0.7; ...
    T0=200;
[x(14,:), y(14,:), vf(14), Ts(14)] = flashPH( P(14), h(14), x0, y0, z(14,:), ...
    T0, VF0 );

%QHA_02 second side
z(15,:)=z(14,:);
n(15)=n(14);
H(15)=H(14)+QHA_02;
h(15)=H(15)/n(15);
x0=[0.0008 0.0131 0.2486 0.5041]; y0=[0.1134 0.3469 0.4186 0.1138]; VF0=0.92; ...
    T0=200;
[x(15,:), y(15,:), vf(15), Ts(15)] = flashPH( P(15), h(15), x0, y0, z(15,:), ...
    T0, VF0 );

%Q_env
%It is calculated by carrying out an energy balance
Q_env=H(16)-H(15);%This corresponds to the heat from the environment
z(17,:)=z(15,:);
n(17)=n(15);
H(17)=Q_env+H(15);
h(17)=H(17)/n(17);
x0=z(17,1:4); y0=z(17,1:4); VF0=1; T0=260;
[x(17,:), y(17,:), vf(17), Ts(17)] = flashPH( P(17), h(17), x0, y0, z(17,:), ...
    T0, VF0 );
H(17)=n(17)*h(17);
Q_env=H(17)-H(15);

% % Output information

x
y
vf
P
Ts
h
z
n
% Save information to file
filename='SSresults.mat';
save(filename)
```

BOG Flow Routine

```matlab
function [c,ceq] = nfunc( N )
% N Refrigerant flow [Kmol/s]
global T p zf hf Q
% This is a simplified version of the main code designed to calculate the
% required refrigerant flow.

%Initialize variables
P=zeros(1,17); % Pressures [pa]
Ts=zeros(1,17); % Stream temperature [K]
n=ones(1,17); % Flows in [kmol/s]
h=zeros(1,17); % Molar enthalpy [Kj/kmol]
vf=zeros(1,17); % Vapor Fraction
H=zeros(1,17); % Total Enthalpy [Kj]
x=zeros(17,5); %Liquid Composition
y=zeros(17,5); %Vapor Composition
z=zeros(17,5); %Total Composition


%Set the pressures for the whole system
lp=[15 14 12 5 13 10 11 16 17];
hp=[1 2 3 4 6 7 8 9];
P(lp)=3e5;
P(hp)=20e5;

%Set known compositions

z(1,:)= [0.1074 0.3292 0.4096 0.1345 0.0193];
z(2,:)= [0.1074 0.3292 0.4096 0.1345 0.0193];
z(16,:)= [0.1074 0.3292 0.4096 0.1345 0.0193];

%Vapor
y(1,:)= [0.1074 0.3292 0.4096 0.1345 0.0193];
y(16,:)= [0.1074 0.3292 0.4096 0.1345 0.0193];

% Set known flows
n=N*n;

%Set known Temperatures
Ts(16)=-33+273.15;%Modified to make it vapour only
Ts(2)=-25+273.15;
Ts(6)=273.15-79;
Ts(7)=273.15-146.6;
Ts(8)=Ts(7);
Ts(9)=Ts(7);
Ts(13)=273.15-101.5;

%Set known vapor fractions
vf(16)=1;
vf(1)=1;
vf(3)=1;
vf(4)=0;

%Compressor
[Ts(1),Wc,h(1),h(16)]= compressor( P(16),Ts(16),P(1),z(16,:),n(1));
H(1)=n(1)*h(1);
H(16)=n(16)*h(16);

%Flash calculation
x0=[0.0080 0.0860 0.4590 0.3450];
y0=[0.1620 0.4570 0.3280 0.1840];
VF0=0.65;
[ x(2,:), y(2,:),vf(2), h(4), h(3)] = flashTP(Ts(2),P(2), x0,y0,z(2,:),VF0);
```

```
h(2)=h(3)*vf(2)+h(4)*(1-vf(3));

%Assign flows and compositions
z(3,:)=y(2,:);
z(4,:)=x(2,:);
n(3)=n(2)*vf(2);
n(4)=n(2)*(1-vf(2));
H(3)=n(3)*h(3);
H(4)=n(4)*h(4);
Ts(3)=Ts(2);
Ts(4)=Ts(2);

%Heat exchanger HA-01
%By assuming adiabatic flash
H(2)=H(3)+H(4);
QHA_01=H(1)-H(2);

%Valve FCV-01
%Isoenthalpic Valve
h(5)=h(4);
n(5)=n(4);
z(5,:)=z(4,:);
x0=[0.1 0.1 0.413 0.437]; y0=[0.032 0.296 0.588 0.079]; VF0=0.26; T0=200;
[x(5,:), y(5,:), vf(5), Ts(5)] = flashPH( P(5), h(5), x0, y0, z(5,:), T0, VF0 );
H(5)=n(5)*h(5);


%Heat exchanger HA-02 first side
%Calculate stream 6
z(6,:)=z(3,:);
n(6)=n(3);
x0=[0.02 0.26 0.59 0.11];y0=[0.28 0.63 0.0 0.001]; VF0=0.5;
[ x(6,:), y(6,:),vf(6), hl, hv] = flashTP(Ts(6),P(6), x0,y0,z(6,:),VF0);
h(6)=hv*vf(6)+hl*(1-vf(6));
H(6)=h(6)*n(6);
QHA_02=H(3)-H(6);

%Calculate HA-04 first side
%Calculate stream 7
z(7,:)=z(6,:);
n(7)=n(6);
vf(7)=0; % Assumed liquid
[Zv,phiv,h(7),Vv]= srks(z(7,:),Ts(7),P(7),1);
H(7)=h(7)*n(7);
QHA_04=H(6)-H(7);

%Split
Sf=0.5;
z(8,:)=z(7,:);
z(9,:)=z(7,:);
vf(8)=vf(7);
vf(9)=vf(7);
n(8)=n(7)*Sf;
n(9)=n(7)*(1-Sf);
h(8)=h(7);
h(9)=h(7);
H(8)=h(8)*n(8);
H(9)=h(9)*n(9);
Ts(8)=Ts(7);
Ts(9)=Ts(7);

%FCV-04 // FCV-05
```

```
z(10,:)=z(8,:);
n(10)=n(8);
h(10)=h(8);
x0=[0.045 0.5 0.38 0.06]; y0=[0.75 0.24 0.01 0]; VF0=0.1; T0=100;
[x(10,:), y(10,:), vf(10), Ts(10)] = flashPH( P(10), h(10), x0, y0, z(10,:), ...
    T0, VF0 );
H(10)=n(10)*h(10);
%Stream 11
z(11,:)=z(9,:);
n(11)=n(9);
h(11)=h(9);
Ts(11)=Ts(10);
x(11,:)=x(10,:);
y(11,:)=y(10,:);
vf(11)=vf(10);
H(11)=n(11)*h(11);

%HA-04 Second side
n(12)=n(10);
H(12)=H(10)+QHA_04;
h(12)=H(12)/n(12);
z(12,:)=z(10,:);
x0=z(12,1:4); y0=x0; VF0=1; T0=200;
[x(12,:), y(12,:), vf(12), Ts(12)] = flashPH( P(12), h(12), x0, y0, z(12,:), ...
    T0, VF0 );


%HA-07 Simplified
%Global energy balance
%This energy is adjusted to the requirements of the natural gas by modifying ...
    the refrigerant flow.
%These requirements are in BOG.m.
z(13,:)=z(11,:);
n(13)=n(11);
x0=[0.0016 0.0786 0.7187 0.1879]; y0=[0.1420 0.7233 0.1327 0.0021];
VF0=0.7;
[ x(13,:), y(13,:),vf(13),hl, hv] = flashTP(Ts(13),P(13), x0,y0,z(13,:),VF0);
h(13)=vf(13)*hv+(1-vf(13))*hl;
H(13)=h(13)*n(13);
QHA_07=H(13)-H(11);

%Import the value Q from the needs of the BOG
errq=abs(Q+QHA_07); %Calculates error, and sets it for fmincon calculation.


c=[]; ceq=[errq];
end
```

Exchanger Sizing

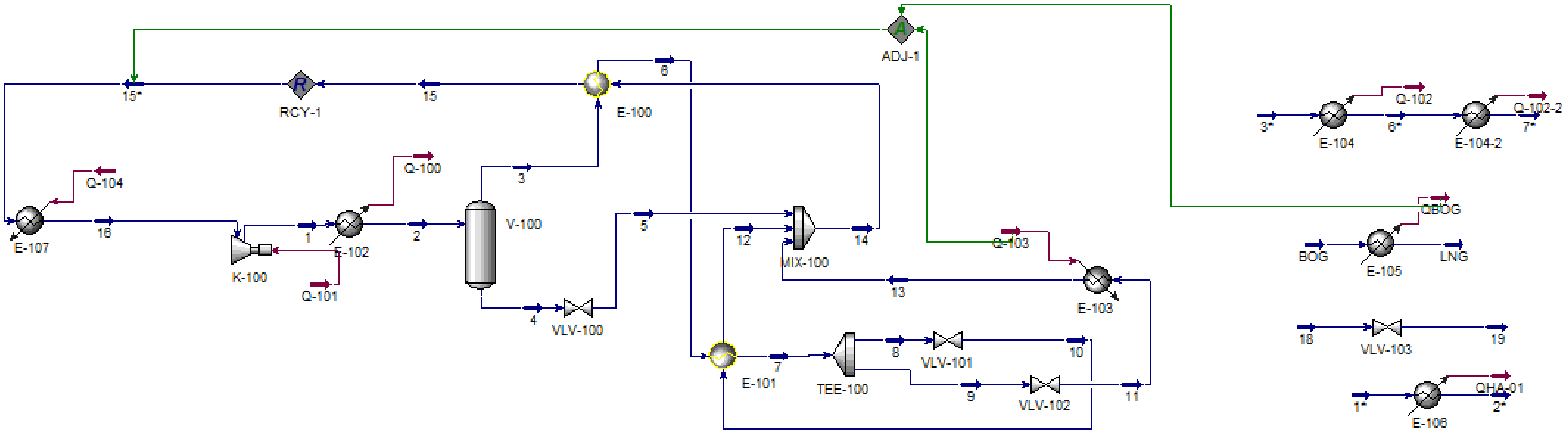
```
% %Heat exchanger sizing

%This script sizes all the heat exchangers based on the information from
%the main steady state simulation.
filename='SSresults.mat';
load (filename)

```

```
% Desuperheater
% Assumes boiling Ethylene as refrigerant (T= 169.5 K)
Tethylene=169.5;
[ UA_HA_01 ] = hxsizing( QHA_01, Ts(1), Ts(2), Tethylene, Tethylene )

% HA-02
[ UA_HA_02 ] = hxsizing( QHA_02, Ts(3), Ts(6), Ts(14), Ts(15) )

% HA-04
[ UA_HA_04 ] = hxsizing( QHA_04, Ts(6), Ts(7), Ts(10), Ts(12) )

% HA-07
%BOG Temperatures from BOG.m
TBOGin = -32 + 273.15; %Inlet temperature [K]
TBOGout = -154 +273.15; %Outlet temperature [K]

[ UA_HA_07 ] = hxsizing( QHA_07, Ts(11), Ts(13), TBOGin, TBOGout )
```

Appendix D

Hysys Model

D.1 Process Flow Diagram Hysys


ADJ-1
6
15*
RCY-1
15
E-100
Q-104
Q-100
3
V-100
E-107
16
1
E-102
2
K-100
Q-101
4
VLV-100
5
12
14
MIX-100
Q-103
13
E-103
7
E-101
TEE-100
8
VLV-101
10
9
VLV-102
11
3*
E-104
Q-102
6*
E-104-2
Q-102-2
7*
QBOG
BOG
E-105
LNG
18
VLV-103
19
1*
E-106
QHA-01
2*