MASTER'S THESIS

Optimization of Distillation Systems

Juliane Ritter

Supervisors:

Prof. Dr. Karl-Heinz Küfer

Dr. Michael Bortz

Dr. Jan Schwientek

Department of Mathematics

TU Kaiserslautern

Department of Optimization

Fraunhofer ITWM

11th June 2018

Declaration

I herewith declare that I wrote and composed the master's thesis at hand independently. I did not use any other sources, figures or resources than the ones stated in the bibliography, be they printed sources or sources off the internet. I marked all passages and sentences in my work that were taken from other sources clearly as such and named the exact source.
Furthermore I declare that - to my best knowledge - this work has never before been submitted by me or somebody else at this or any other university.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A chemical plant uses chemical processes in order to transform feedstock materials into products. Chemical processes can be divided roughly into two categories: *Transformations* use chemical reactions or biological processes to convert available compounds into new ones, while *separations* exploit differences in the chemical or physical properties of the constituents of a mixture to produce two or more mixtures with different compositions.
In Germany the chemical industry accounts for about 10% of the overall primary energy demand, as can be seen in Figure 1.1. It is estimated that over 40% of the energy demand of a chemical plant is due to separation processes [1]. Therefore improving the efficiency of separation processes is an important step in reducing the overall energy demand.

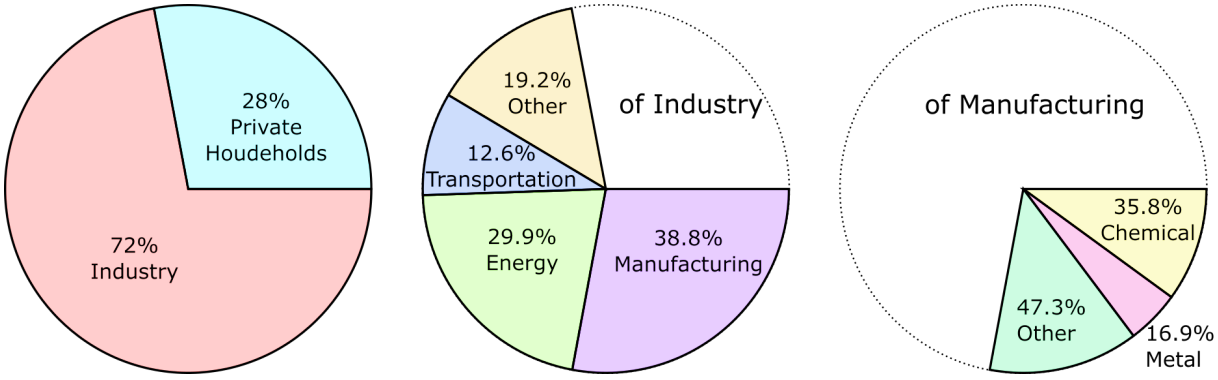


Figure 1.1: Distribution of the primary energy demand in Germany in 2013. Data from [2].

In this work, we focus exclusively on the process of distillation, which is used for around 95% of all separations tasks in the chemical industry [1]. Large-scale industrial distillation is typically performed as a continuous distillation in a distillation column. Often, in order to separate a mixture of multiple components into products of desired purity, a system of distillation columns is needed. The number of potential structures grows rapidly in the number of components present.
The goal of this work is to examine methods for finding the optimal distillation system for a given separation task. To achieve this, we follow the following steps:
In Part I we lay the foundations: We explain the thermodynamic principles behind the method of distillation and derive equations for modelling a distillation column. Further, we introduce some shortcut methods for modelling a distillation column based on suitable simplifications. Additionally, we introduce some basic definitions and results of mathematical optimization and examine different

types of optimization problems and their properties and typical algorithms used in solvers.
In Part II we review literature that has treated the problem of finding the optimal distillation system. We first compare different definitions of a feasible distillation system and give results that justify restricting the feasible set to a specific type. We then examine formulations for describing this search space mathematically. Further, we briefly describe and analyse different algorithms that have been developed to search for the optimal distillation system.
In Part III we describe the two approaches we have developed and tested in this work:
The first approach combines the rigorous models for distillation, introduced in Part I, with the superstructure formulation for the feasible set of distillation systems, discussed in Part II.
The second approach is based on the Underwood shortcut model for distillation, introduced in Part I, and a matrix-based formulation for the feasible set of distillation systems, examined in Part II.
Firstly, we give a detailed derivation of all variables and constraints arising from both approaches. Then, we discuss the technical framework in which they were implemented and solved numerically. Finally, we compare and discuss the results obtained in both cases for multiple test scenarios and using a variety of solvers.
We finish this thesis with an outlook on further problems that arise from this work and may be examined in the future.

Part I

Foundations

Chapter 2

Distillation

In this chapter, we will first discuss the thermodynamical theory behind distillation: the vapour-liquid equilibrium. We further examine how to model its properties at different levels of simplification. Then we describe the workings of a distillation column and discuss different approaches to modelling it, namely the rigorous MESH-equations and multiple shortcut methods such as Fenske or Underwood.

2.1 Vapour-liquid equilibrium

In the following, we will consider a system comprised of a liquid and a gas phase. It contains N_C distinct chemical species, called *components* and labelled $1, \dots, N_C$. We assume that the phases are in equilibrium. Then we can describe the state of the system by the following properties:

- temperature T in K
- pressure P in kPa
- molar fraction of component c in the liquid phase x_c, $c = 1, \dots, N_C$
- molar fraction of component c in the gas phase y_c, $c = 1, \dots, N_C$.

Note that the by definition of the molar fraction, the so called *summation equations* hold:

$$\sum_{c=1}^{N_C} x_c = 1, \quad \sum_{c=1}^{N_C} y_c = 1. \tag{2.1}$$

Since in the majority of industrial distillation the pressure is approximately constant [3], we consider P to be fixed in the following.

2.1.1 Txy-diagram and xy-diagram

For binary mixtures ($N_C = 2$), the relation between temperature and composition of liquid and gas phase at equilibrium can be represented as a two-dimensional graph called *Txy-diagram*. An example is given in Figure 2.1. Note that, in general, this diagram depends on the (fixed) value of P.
For each temperature T on the vertical axis, there exists a unique pair of compositions (x_1, x_2) and (y_1, y_2) such that the phases are in equilibrium. These are marked as points on the boiling and dew point curve, respectively.

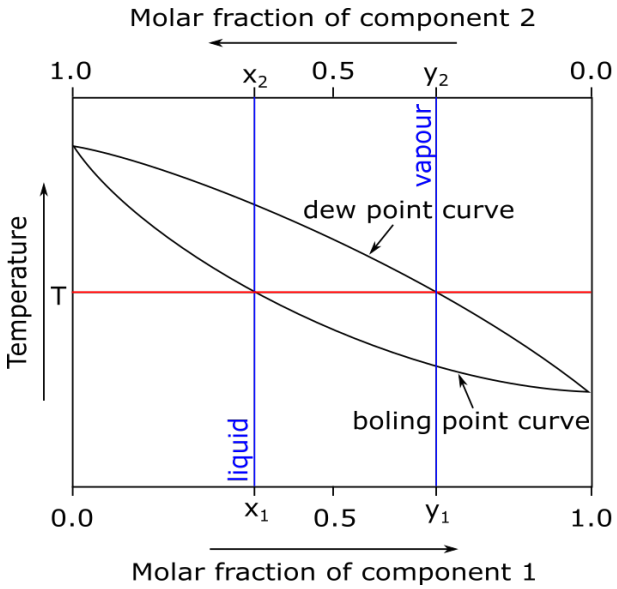


Figure 2.1: Txy-diagram of a zeotropic mixture. Adapted from [4].

Another interpretation is given as follows: a liquid of composition (x_1, x_2) starts to boil at temperature T to form vapour of composition (y_1, y_2). Conversely, a gas of composition (y_1, y_2) starts to dew at temperature T to form liquid of composition (x_1, x_2).
We see that boiling and dew point curve only intersect at the pure components, i.e. the compositions $(1, 0)$ and $(0, 1)$. Note, that the corresponding temperatures are exactly the boiling points of the pure components 1 and 2, respectively. Therefore, when a liquid of any composition starts to boil, it always forms vapour that is richer in the component with lower boiling point, here component 1. A mixture of components with this property is called *zeotropic*.

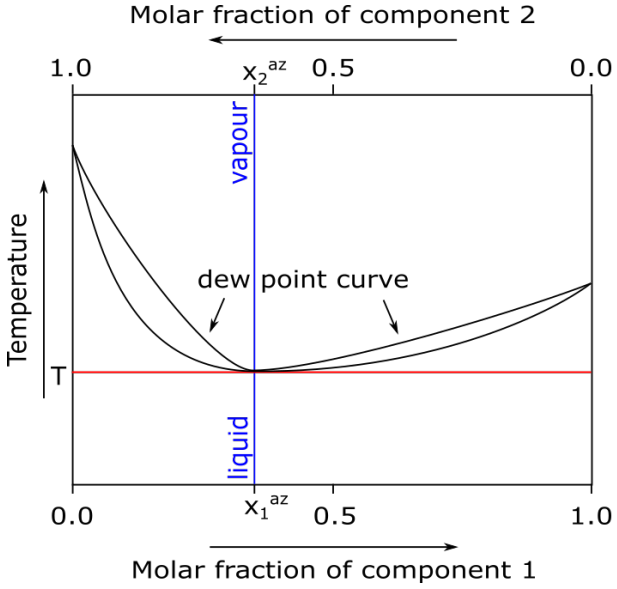


Figure 2.2: Txy-diagram of an azeotropic mixture.

Figure 2.2 shows the Txy-diagram of an *azeotropic* mixture. In this case, there exists a non-trivial composition (x_1^{az}, x_2^{az}) at which boiling and dew point curve intersect. At this point there occurs no change upon boiling, since $(y_1^{az}, y_2^{az}) = (x_1^{az}, x_2^{az})$.

Further, there are two sections with different behaviours: For low concentrations of component 1 ($x_1 < x_1^{az}$), formed vapour is enriched in component 1, which is the component with a lower boiling point. But for high concentrations of component 1 ($x_1 > x_1^{az}$), formed vapour is enriched in component 2. Just by the means of vapourisation and condensation, it is not possible to move between those two sections of compositions.
It is sometimes more useful to plot the compositions (x_1, x_2) and (y_1, y_2) directly against each other. The so called *xy-diagrams* corresponding to the previous examples of Txy-diagrams are shown in Figure 2.3.

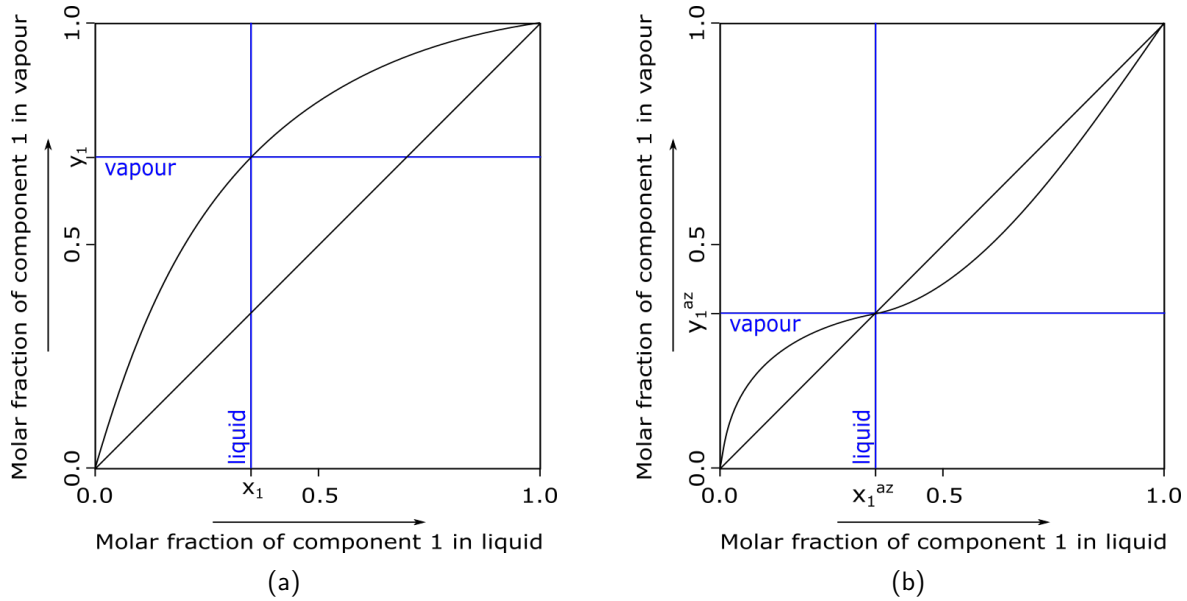


Figure 2.3: xy-diagrams of (a) a zeotropic mixture, corresponding to Figure 2.1 and (b) an azeotropic mixture, corresponding to Figure 2.2.

2.1.2 Modelling

The vapour-liquid equilibrium data to draw a Txy-diagram for a given mixture at fixed pressure can be obtained by experimentation. Different models have been derived to describe the results systematically, now again for the general case of N_C components:
In general, it can be assumed that the gas phase behaves like an ideal gas. Then *Dalton's law* holds:

$$P = \sum_{c=1}^{N_C} P_c$$

where P_c is the *partial vapour pressure* of component c. Together with the summation equation (2.1) for y this yields (assuming independence of the components):

$$P_c = P\ y_c, \quad c = 1, \ldots, N_C. \tag{2.2}$$

2.1.2.1 Ideal mixture

An *ideal mixture* is a liquid, in which the mean strength of interactions is the same between all molecules in the solution. This is a good approximation for mixtures of components with similar

physical properties, for example mixtures of alkanes. Under assumption of ideality of the liquid phase, *Raoult's law* holds:

$$P_c = P_c^0 \; x_c, \quad c = 1, \ldots, N_C \tag{2.3}$$

where P_c^0 is the *vapour pressure* of component c.
Combining this result with equation (2.2) we obtain

$$y_c = \frac{P_c^0}{p} \; x_c, \quad c = 1, \ldots, N_C. \tag{2.4}$$

Note that the factor P_c^0 is both dependent on the component c and the temperature T.

2.1.2.2 Constant relative volatility

We define the *volatility* of component $c \in \{1, \ldots, N_C\}$, as

$$v_c = \frac{P_c}{x_c}.$$

Further, the *relative volatility* of component d to c with $d, c \in \{1, \ldots, N_C\}$ is defined as

$$\alpha_{d,c} = \frac{v_d}{v_c} = \frac{P_d \; x_c}{x_d \; P_c}.$$

Using equation (2.2), this implies

$$\alpha_{d,c} = \frac{y_d \; x_c}{x_d \; y_c} = \frac{\frac{y_d}{y_c}}{\frac{x_d}{x_c}} \tag{2.5}$$

and therefore

$$\frac{1}{y_c} = \frac{\sum_{d=1}^{N_C} y_d}{y_c} = \sum_{d=1}^{N_C} \frac{y_d}{y_c} = \sum_{d=1}^{N_C} \alpha_{d,c} \; \frac{x_d}{x_c} = \frac{\sum_{d=1}^{N_C} \alpha_{d,c} \; x_d}{x_c}.$$

This finally yields:

$$y_c = \frac{x_c}{\sum_{d=1}^{N_C} \alpha_{d,c} \; x_d}. \tag{2.6}$$

Note that the coefficients $\alpha_{d,c}$ are again dependent on the components d and c and the temperature T. Specifically, under the assumption of an ideal mixture, we obtain by inserting equation (2.3) into the definition

$$\alpha_{d,c} = \frac{P_d^0}{P_c^0}.$$

For some mixtures (specifically zeotropic mixtures), the relative volatilities are approximately constant over the working temperature range of the distillation. Under this assumption of *constant relative volatility*, equation (2.6) can be used to calculate the compositions at equilibrium independent of the temperature.

In this case, it is common to abbreviate $\alpha_c = \alpha_{c,N_C}$ where the components are sorted by decreasing volatilities $\alpha_1 > \alpha_2 > \ldots > \alpha_{N_C - 1} > \alpha_{N_C} = 1$. Since $\alpha_{d,c} = \frac{\alpha_{d,N_C}}{\alpha_{c,N_C}}$, equation (2.6) becomes

$$y_c = \frac{\alpha_c \; x_c}{\sum_{d=1}^{N_C} \alpha_d \; x_d}. \tag{2.7}$$

2.1.2.3 Non-Ideal mixture

For mixtures that can not be approximately treated as ideal, Raoult's law is modified to

$$P_c = \gamma_c \, P_c^0 \, x_c, \quad c = 1, \ldots, N_C \tag{2.8}$$

where γ_c is the *activity coefficient* of component c.
The activity coefficient can be interpreted as follows: If $\gamma_c < 1$ the modified Raoult's law yields a lower partial vapour pressure P_c than in the case of an ideal mixture. This means, the adhesive forces between molecules of different components are stronger than the cohesive forces between molecules of the same component. Conversely, $\gamma_c > 1$ models the case that the cohesive forces are stronger than the adhesive forces.
By using equation (2.2) we obtain, analogously to equation (2.4):

$$y_c = \gamma_c \, \frac{P_c^0}{P} \, x_c, \quad c = 1, \ldots, N_C. \tag{2.9}$$

In this case, we have an additional factor γ_c that is not only dependent on the component c and the temperature T, but also the composition x_d, $d = 1, \ldots, N_C$.

2.1.2.4 Relative volatility, vapour pressure and activity coefficient

As previously mentioned, some of the coefficients, appearing in the introduced models for the vapour-liquid equilibrium, again depend on other variables. Further, all of the coefficients depend on the mixture considered.
We explain how to obtain completely described models in the following:
For the model based on the assumption of constant relative volatilities, see equation (2.6), it suffices to fit the values of α_c for the components of the considered mixture to experimental data. We obtained the values for the mixtures discussed in Part III from literature.
For the models based on Raoult's law, see equations (2.9) and (2.4), we need an additional model to describe the temperature dependence of the vapour pressure P_c^0. We use the DIPPR-101 model which is established in industry applications:

$$P_c^0 = \exp(A_c + B_c/T + C_c \ln(T) + D_c T^{E_c}) \text{ Pa} \tag{2.10}$$

with parameters A_c, B_c, C_c, D_c, E_c.
For the general non-ideal model, see equation (2.9), we further need a model to describe the behaviour of the activity coefficients γ_c. We use the NRTL model:

$$\gamma_c = \exp\left(\frac{\sum_d x_d G_{dc} \tau_{dc}}{\sum_d x_d G_{dc}} + \sum_d \frac{x_d G_{cd}}{\sum_e x_e G_{ed}} \left(\tau_{cd} - \frac{\sum_e x_e G_{ed} \tau_{ed}}{\sum_e x_e G_{ed}}\right)\right) \tag{2.11}$$

where

$$\tau_{cd} = A_{cd} + \frac{B_{cd}}{T}$$
$$G_{cd} = \exp(-C_{cd} \tau_{cd}).$$

with parameters A_{cd}, B_{cd}, C_{cd}. We observe that the special case of an ideal mixture, which corresponds to $\gamma_c = 1,\ c = 1, \ldots, N_C$ in the NRTL model, can be modelled by setting $A_{cd} = 0, B_{cd} = 0, C_{cd} = 0,\ c, d = 1, \ldots, C$.
For both the DIPPR-101 and the NRTL model, we obtained the values of the parameters for the mixtures discussed in Part III from a BASF database, which uses experimental data from many different sources to fit them.

2.1.3 Summary

The vapour formed when boiling a liquid zeotropic mixture is enriched in the components with lower boiling points. Distillation uses this fact to separate mixtures. Since this behaviour is not guaranteed for azeotropic mixtures, we will exclude them from our work.
The composition of the vapour in equilibrium with a liquid of given composition, temperature and pressure can be obtained by the equation (2.9) with the additional property equations (2.10) and (2.11). Under the assumption of an ideal mixture, they reduce to equations (2.4) and (2.10). Under the assumption of constant relative volatility we only need to consider the single equation (2.7).

2.2 Distillation column

A distillation column is a cylindrical structure which facilitates the separation of mixtures by repeated vapourisation and condensation. A schematic representation is given in Figure 2.4.
Here, the column is divided into sections by bubble-cap-trays, a more detailed picture of which is given in Figure 2.5. The bubble caps permit the upward flow of vapour but prevent the downward flow of liquid. The liquid is collected in a hold-up and is moved along the tray to the weir, where overflow of the tray flows down. Alternative tray designs such as sieve trays or valve trays work analogously. The aim of all three is to provide good mixing of vapour and liquid while minimizing pressure drop along the column. Other factors such as cost, operability and maintainability are weighed in when deciding between the options. [6]
In some industrial distillations the distillation trays are replaced by packing. This option often provides lower pressure drop but is more restricted in terms of feasible liquid and vapour flow rates. [7]
There exists some intermediate tray where the feed is introduced. In the pictured example, the feed is liquid, but it might as well be (partly) vapour. The portion of the column above the feed tray is called *rectification* section, the portion below *stripping* section.

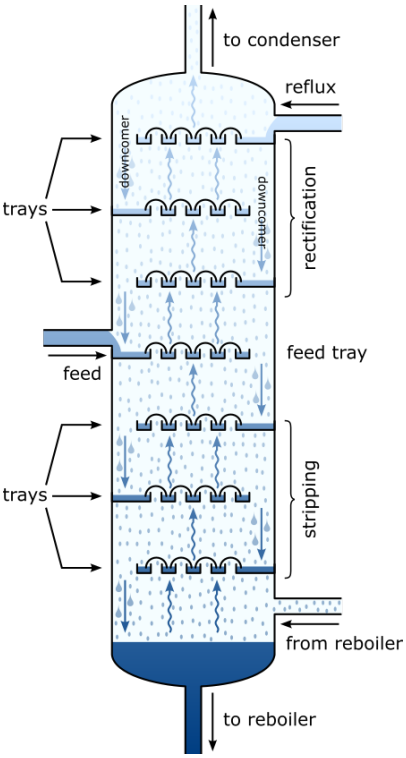


Figure 2.4: Schematic representation of a distillation column. Adapted from [5].

On the bottom of the column there is a liquid hold up. The outgoing liquid stream is split into a *bottom product* stream and a stream which leads to an external reboiler. The produced vapour is transferred back into the column, below the lowest tray.
Analogously, on the top of the column, there is an outgoing vapour stream which is split up into a *distillate product* stream and a stream which leads to an external condenser. The produced liquid is transferred back to the column at the highest tray location. Often, the distillate product stream is also fed through the condenser to obtain both product streams as liquids.

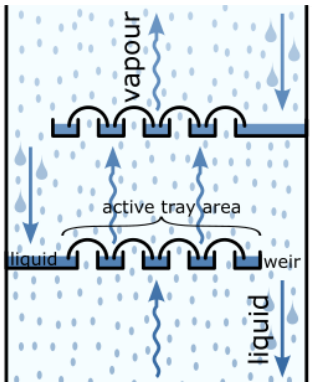


Figure 2.5: Schematic representation of the operating principle of bubble cap trays.

2.2.1 Modelling

2.2.1.1 Equilibrium stage

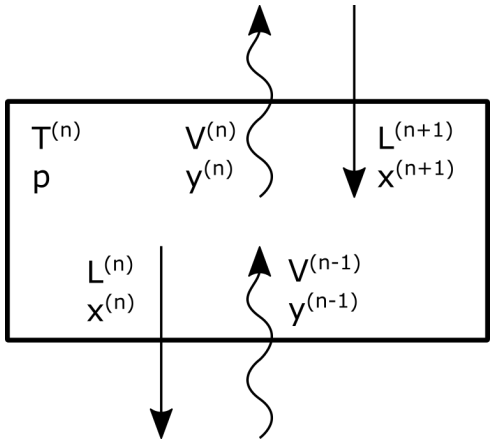


Figure 2.6: Schematic representation of an equilibrium stage.

An equilibrium stage models a tray within a distillation column, under the assumption that the vapour and liquid streams leaving the tray are in equilibrium. A schematic representation with the introduced variables is given in Figure 2.6.
In reality, the degree of enrichment of the vapour (in the components with lower boiling point) is smaller. This means, a real-world distillation column will, in general, need more trays than its model counterpart in order to facilitate the same separation task.
We consider the general case of a non-ideal mixture of N_C components, therefore the vapour-liquid equilibrium is described by (compare (2.9)):

$$0 = P\ y_c^{(n)} - \gamma_c(T^{(n)}, x^{(n)})\ P_c^0(T^{(n)})\ x_c^{(n)}, \quad c = 1, \dots, N_C. \tag{2.12}$$

The law of conservation of mass implies, that the incoming and outgoing streams must be balanced in their molarity in each component, therefore:

$$0 = V^{(n-1)}\ y_c^{(n-1)} + L^{(n+1)}\ x_c^{(n+1)} - V^{(n)}\ y_c^{(n)} - L^{(n)}\ x_c^{(n)}, \quad c = 1, \dots, N_C. \tag{2.13}$$

Analogously, the law of conservation of energy implies, that the incoming and outgoing streams must

be balanced in their enthalpy (neglecting heat losses and heat of mixing), therefore:

$$0 = \sum_{c=1}^{N_C} (V^{(n-1)}\ y_c^{(n-1)}\ H_c(T^{(n-1)}) + L^{(n+1)}\ x_c^{(n+1)}\ h_c(T^{(n+1)}) - V^{(n)}\ y_c^{(n)}\ H_c(T^{(n)}) - L^{(n)}\ x_c^{(n)}\ h_c(T^{(n)})) \tag{2.14}$$

where $H_c(T)$ and $h_c(T)$ are the enthalpy of vapour and liquid, respectively, of one mole of component c at temperature T.

Additionally, the following summation equations (compare (2.1)) hold:

$$0 = 1 - \sum_{c=1}^{N_C} y_c^{(n-1)} = 1 - \sum_{c=1}^{N_C} x_c^{(n+1)} = 1 - \sum_{c=1}^{N_C} y_c^{(n)} = 1 - \sum_{c=1}^{N_C} x_c^{(n)}. \tag{2.15}$$

These equations are called *MESH-equations*, which is an abbreviation for *M*aterial balances, *E*quilibrium equations, *S*ummation equations and *H*eat balances [8]. In the following we will arrange them in this order.

2.2.1.2 Standard distillation column

We model a standard distillation column as a concatenation of N_S equilibrium stages with connecting streams. A schematic representation with the introduced variables is given in Figure 2.7. At some intermediate stage n_F the feed of flow rate F and composition x^F is introduced as liquid at boiling point. The distillate and bottom products of molar flow rates D and B and compositions x^D and x^B, respectively, are both obtained as liquids.

Note that we model a total condenser, therefore $y^{(N_S)} = x^D = x^{CO}$ which means the last two variables are redundant. Analogously, since we model a total reboiler $x^{(1)} = x^B = y^{RE}$, which means the last two variables are again redundant. Furthermore CO is redundant by $CO = V^{(N_S)} - D$ and RE by $RE = L^{(1)} - B$. We will not use the variables that were identified as redundant in the following equations.

Generalizing the equations derived for the ideal stage yields:

Material balances (compare (2.13)):

$$\begin{aligned}
0 &= L^{(2)}\ x_c^{(2)} - V^{(1)}\ y_c^{(1)} - B\ x_c^{(1)} \\
0 &= V^{(n-1)}\ y_c^{(n-1)} + L^{(n+1)}\ x_c^{(n+1)} - V^{(n)}\ y_c^{(n)} - L^{(n)}\ x_c^{(n)} \\
0 &= V^{(n_F-1)}\ y_c^{(n_F-1)} + L^{(n_F+1)}\ x_c^{(n_F+1)} - V^{(n_F)}\ y_c^{(n_F)} - L^{(n_F)}\ x_c^{(n_F)} + F\ x_c^F \\
0 &= V^{(N_S-1)}\ y_c^{(N_S-1)} - L^{(N_S)}\ x_c^{(N_S)} - D\ y_c^{(N_S)}
\end{aligned} \tag{2.16}$$

with $c = 1, \ldots, N_C$ and $n = 2, \ldots, n_F - 1, n_F + 1, \ldots, N_S - 1$.

Equilibrium equations (compare (2.12)):

$$0 = P\ y_c^{(n)} - \gamma_c(T^{(n)}, x^{(n)})\ P_c^0(T^{(n)})\ x_c^{(n)} \tag{2.17}$$

with $c = 1, \ldots, N_C$ and $n = 1, \ldots, N_S$.

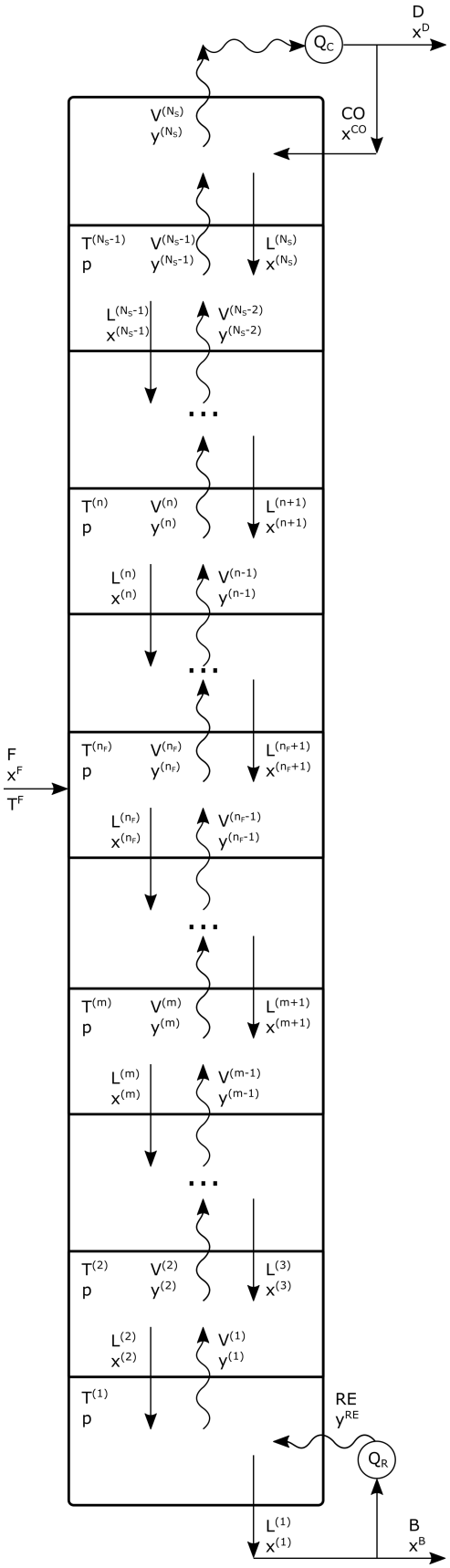


Figure 2.7: Schematic representation of a standard distillation column.

Summation equations (compare (2.15)):

$$\begin{aligned} 0 &= 1 - \sum_{c=1}^{N_C} x_c^{(n)} \\ 0 &= 1 - \sum_{c=1}^{N_C} y_c^{(n)} \end{aligned} \tag{2.18}$$

with $n = 1, \ldots, N_S$.

Heat balances (compare (2.14)):

$$\begin{aligned} 0 = &\sum_{c=1}^{N_C} \Big[L^{(2)} x_c^{(2)} h_c(T^{(2)}) - V^{(1)} y_c^{(1)} H_c(T^{(1)}) - \ldots \\ &\ldots B x_c^{(1)} h_c(T^{(1)} \Big] + Q_R \\ 0 = &\sum_{c=1}^{N_C} \Big[V^{(n-1)} y_c^{(n-1)} H_c(T^{(n-1)} + L^{(n+1)} x_c^{(n+1)} h_c(T^{(n+1)}) - \ldots \\ &\ldots V^{(n)} y_c^{(n)} H_c(T^{(n)}) - L^{(n)} x_c^{(n)} h_c(T^{(n)}) \Big] \\ 0 = &\sum_{c=1}^{N_C} \Big[V^{(n_F-1)} y_c^{(n_F-1)} H_c(T^{(n_F-1)} + L^{(n_F+1)} x_c^{(n_F+1)} h_c(T^{(n_F+1)}) - \ldots \\ &\ldots V^{(n_F)} y_c^{(n_F)} H_c(T^{(n_F)}) - L^{(n_F)} x_c^{(n_F)} h_c(T^{(n_F)}) + F x_c^F h_c(T^F) \Big] \\ 0 = &\sum_{c=1}^{N_C} \Big[V^{(N_S-1)} y_c^{(N_S-1)} H_c(T^{(N_S-1)}) - \ldots \\ &\ldots L^{(N_S)} x_c^{(N_S)} h_c(T^{(N_S)}) - D y_c^{(N_S)} h_c(T^{(N_S)}) \Big] + Q_C \end{aligned} \tag{2.19}$$

with $n = 2, \ldots, n_F - 1, n_F + 1, N_S - 1$.

To model the fact, that the feed is liquid at its boiling point, we also introduce the equation:

$$0 = P - \sum_{c=1}^{N_C} \gamma_c(T^F, x^F)\ P_c^0(T^F)\ x_c^F. \tag{2.20}$$

In case of a partly vapour feed, we need to introduce the vapour and liquid flow rates F_v and F_l and their corresponding compositions y^{F_v} and x^{F_l}. Since the phases must be in equilibrium, equation (2.20) is adapted to:

$$0 = P\ y_c^{F_v} - \gamma_c(T^F, x^{F_l})\ P_c^0(T^F)\ x_c^{F_l}$$

with $c = 1, \ldots, N_C$. In the material balance, we replace $F\ x_c^F$ by $F_v\ y_c^{F_v} + F_l\ x_c^{F_l}$ and in the heat balance $F\ x_c^F\ h_c(T^F)$ by $F_v\ y_c^{F_v}\ H_c(T^F) + F_l\ x_c^{F_l}\ h_c(T^F)$.

2.2.1.3 Enthalpy

To complete the model of a standard distillation column, we describe the temperature dependence of the enthalpy with models widely used in industrial applications:
Enthalpy of vapour:

$$H_c(T) = A_c T + B_c \frac{C_c}{\tanh(\frac{C_c}{T})} + D_c E_c \tanh(\frac{E_c}{T}) - (A_c 298 + B_c \frac{C_c}{\tanh(\frac{C_c}{298})} + D_c E_c \tanh(\frac{E_c}{298})) \frac{\mathsf{J}}{\mathsf{kmol}} \tag{2.21}$$

with parameters A_c, B_c, C_c, D_c, E_c.
Enthalpy of vapourisation:

$$H_c^{VL}(T) = A_c(1 - \frac{T}{B_c})^{C_c + \frac{T}{B_c}(D_c + E_c \frac{T}{B_c})} \frac{\mathsf{J}}{\mathsf{kmol}}$$

with parameters A_c, B_c, C_c, D_c, E_c.
Enthalpy of liquid:

$$h_c(T) = H_c(T) - H1VL_c(T) \frac{\mathsf{J}}{\mathsf{kmol}} \tag{2.22}$$

We obtain the values of the parameters for the mixtures discussed in Part III from a BASF database.

2.2.2 Shortcut methods

The MESH-equations (2.16)-(2.20) together with the property equations (2.10), (2.11) and (2.21), (2.22) give a rigorous model of the physical processes that govern a distillation column. However, the highly non-linear temperature dependences of vapour pressures, activity coefficients and vapour/liquid enthalpies complicate numerical calculations.
In the following, we will introduce some methods that have been developed to determine certain properties of a distillation column under simplifying assumptions resulting in simplified equations.
We assume that the N_C components of our zeotropic mixture are ordered by increasing boiling point. Note, that in case of constant relative volatilities, this aligns with ordering by decreasing relative volatility. Then, by the properties of the vapour-liquid equilibrium, the vapour leaving each equilibrium stage is enriched in the first components. This means, we can only eliminate the last (consecutive) c components from our distillate product and the first (consecutive) d components from our bottom product. Note, that it must hold $c + d \leq N_C$, since no components can vanish.
Therefore, we can abbreviate every separation performed by a single distillation column, also called *split*, as a line separating sets of consecutive components, with no missing but potentially overlapping components. For example, $12|23$ describes the separation of the feed 123 into distillate product 12 and bottom product 23.
In general, we will assume that a separation is not perfect, i.e. there is a small amount of impurity in the components neighbouring the split. For example, in case of the split $12|23$, there is a small, but non-zero, concentration of component 1 in the bottom product and of component 3 in the distillate product. In some of the following shortcut models the separation is completely described by specifying the concentrations of these *key components*. We call the component lighter than the lightest component in the bottom product (according to the perfect split) *light key*, the one heavier than the heaviest component in the distillate product *heavy key*.

2.2.2.1 Constant molar overflow

By abbreviating

$$H^{(n)} := \sum_{c=1}^{N_C} y_c^{(n)} H_c(T^{(n)})$$
$$h^{(n)} := \sum_{c=1}^{N_C} x_c^{(n)} h_c(T^{(n)})$$

the heat balance for a stage in the rectifying section, $n \in \{n_F + 1, \ldots, N_S - 1\}$, simplifies to

$$0 = V^{(n-1)} H^{(n-1)} + L^{(n+1)} h^{(n+1)} - V^{(n)} H^{(n)} - L^{(n)} h^{(n)}.$$

This implies

$$H^{(n)}(V^{(n)} - V^{(n-1)}) - V^{(n-1)}(H^{(n-1)} - H^{(n)}) = h^{(n+1)}(L^{(n+1)} - L^{(n)}) - L^{(n)}(h^{(n)} - h^{(n+1)})$$

and therefore by using the overall material balance $L^{(n+1)} - L^{(n)} = V^{(n)} - V^{(n-1)}$

$$V^{(n)} - V^{(n-1)} = \frac{V^{(n-1)}(H^{(n-1)} - H^{(n)}) - L^{(n)}(h^{(n)} - h^{(n+1)})}{H^{(n)} - h^{(n+1)}}. \tag{2.23}$$

An analogous derivation can be used for the stripping section.

A very common simplification is the assumption of *constant molar overflow*, which means the vapour and liquid mass flows are constant in each section of the column. By equation (2.23) this assumption is satisfied, if

$$H^{(n)} = const,\ h^{(n)} = const.$$

This is approximately fulfilled if the enthalpy of vapourisation is similar for all components in the mixture and enthalpy changes due to temperature differences between stages are negligible. Note, that the assumption can also be satisfied for varying enthalpies if

$$\frac{H^{(n-1)} - H^{(n)}}{h^{(n)} - h^{(n+1)}} = \frac{L^{(n)}}{V^{(n-1)}}$$

[9, p.217]. Generally, this assumption holds well for mixtures where the components have similar physical properties. Specifically, mixtures that can be modelled by constant relative volatilities also can be assumed to exhibit constant molar overflow. [9]

Let $V^{(r)},\ L^{(r)}$ and $V^{(s)},\ L^{(s)}$ be the mass flows in the rectifying and stripping section, respectively. The material balances (2.16) simplify to:

$$\begin{aligned}
0 &= L^{(s)}\ x_c^{(2)} - V^{(s)}\ y_c^{(1)} - B\ x_c^{(1)} \\
0 &= V^{(s)}\ y_c^{(m-1)} + L^{(s)}\ x_c^{(m+1)} - V^{(s)}\ y_c^{(m)} - L^{(s)}\ x_c^{(m)} \\
0 &= V^{(s)}\ y_c^{(n_F-1)} + L^{(r)}\ x_c^{(n_F+1)} - V^{(s)}\ y_c^{(n_F)} - L^{(r)}\ x_c^{(n_F)} + F\ x_c^F \\
0 &= V^{(r)}\ y_c^{(n-1)} + L^{(r)}\ x_c^{(n+1)} - V^{(r)}\ y_c^{(n)} - L^{(r)}\ x_c^{(n)} \\
0 &= V^{(r)}\ y_c^{(N_S-1)} - L^{(r)}\ x_c^{(N_S)} - D\ y_c^{(N_S)}
\end{aligned}$$

with $c = 1, \ldots, N_C$ and $m = 2, \ldots, n_F - 1$, $n = n_F + 1, \ldots, N_S - 1$.
The equilibrium equations (2.17) and summation equations (2.18) remain unchanged, the heat balances are fulfilled by assumption and can be eliminated.

2.2.2.2 Lewis-Sorel method

We calculate the minimum number of stages needed to reach given product specifications for a fixed feed under the assumption of constant molar overflow.
Let $F,\ x^F,\ D,\ x^D,\ B,\ x^B$ be fixed. Note, that it suffices to know $F,\ x^F,\ D$ and $x_c^D,\ c = 1, \ldots, N_C - 1$, since we can obtain $B = F - D$, $x_{N_C}^D = 1 - \sum_{c=1}^{N_C - 1} x_c^D$ and $x_c^B = (F x_c^F - D x_c^D)/B$, $c = 1, \ldots, N_C$ from the overall material balance and summation equations.
Additionally, we need to fix the *reflux ratio* $R = \frac{L^{(r)}}{D}$. This implies $L^{(r)} = RD$ and $V^{(r)} = (R+1)D$.
Considering the material balance around the rectifying section above stage $n \in \{n_F + 1, \ldots, N_S - 1\}$ gives:

$$\begin{aligned} V^{(r)}\ y_c^{(n)} &= L^{(r)}\ x_c^{(n+1)} + D\ x_c^D \\ \Rightarrow \qquad y_c^{(n)} &= \frac{L^{(r)}}{V^{(r)}}\ x_c^{(n+1)} + \frac{D}{V^{(r)}}\ x_c^D \\ &= \frac{R}{R+1}\ x_c^{(n+1)} + \frac{1}{R+1}\ x_c^D, \quad c = 1, \ldots N_S. \end{aligned} \tag{2.24}$$

This linear relation between the compositions of the liquid leaving a stage and the vapour entering it is called *operating line* of the rectifying section.
To obtain a similar result for the stripping section, we introduce the *thermal quality* q of the feed. There are five cases that can be modelled:

- $q > 1$: The feed is liquid and colder than its boiling point, so some additional vapour is condensed to provide the heat to bring it up to its boiling point.
- $q = 1$: The feed is liquid at its boiling point.
- $0 < q < 1$: The feed is partly liquid, partly vapour, q indicates the fraction that is liquid.
- $q = 0$: The feed is vapour at its dew point.
- $q < 0$: The feed is vapour and warmer than its dew point, so some additional liquid is vaporised by the heat given off to bring it down to its dew point.

Now we can write $L^{(s)} = L^{(r)} + qF = RD + qF$ and $V^{(s)} = RD + qF - B$.
Considering the material balance around the stripping section up to stage $m \in \{2, \ldots, n_F - 1\}$ gives:

$$\begin{aligned} V^{(s)}\ y_c^{(m)} &= L^{(s)}\ x_c^{(m+1)} - B\ x_c^B \\ \Rightarrow \qquad y_c^{(m)} &= \frac{L^{(s)}}{V^{(s)}}\ x_c^{(m+1)} - \frac{B}{V^{(s)}}\ x_c^B \\ &= \frac{S+1}{S}\ x_c^{(m+1)} - \frac{1}{S}\ x_c^B, \quad c = 1, \ldots N_S \end{aligned} \tag{2.25}$$

with $S := \frac{V^{(s)}}{B} = \frac{RD + qF - B}{B}$. This is similarly called the operating line of the stripping section.

For the intersection of both operating lines (x^q, y^q) we have:

$$
\begin{aligned}
& (V^{(s)} - V^{(r)}) y_c^q = (L^{(s)} - L^{(r)}) x_c^q - (B x_c^B + D x_c^D) \\
\Rightarrow \quad & -(1-q) F y_c^q = q F x_x^q - F x_c^F \\
\overset{q \neq 1}{\Rightarrow} \quad & y_c^q = \frac{q}{q-1} x_c^q - \frac{1}{q-1} x_c^F, \qquad c = 1, \dots N_S.
\end{aligned}
$$

This linear relation is called *q-line* and depends solely on the feed composition and its thermal quality. Note that in the case of a boiling liquid feed ($q = 1$), instead of the last transformation we directly obtain $x_c^q = x_c^F,\ c = 1, \dots, N_C$.

We can now calculate the composition on each stage by starting from the top:

We start with $y^{(N_S)} = x^D$ and obtain x^{N_S} from the vapour-liquid equilibrium. Then we calculate $y^{(N_S-1)}$ from (2.24). This can be iterated until we reach a composition that is close to the feed composition. Then we continue analogously, but using (2.25) instead, until we obtain a composition that satisfies our specifications for x^B. The number of iterations needed equals the minimum number of stages.

This procedure is called *Lewis-Sorel* method. In the case of a binary mixture it can be executed graphically and is then often called *McCabe-Thiele* method. An example of its application is given in Figure 2.8.

We use the xy-diagram of the vapour-liquid equilibrium and add vertical lines for the specified compositions of feed, distillate and bottom. Furthermore, we draw the operating line of the rectifying section and the q-line. Then we can draw the operating line for the stripping section by connecting the intersection of rectifying operating line and q-line with the intersection of bottom composition and identity line.

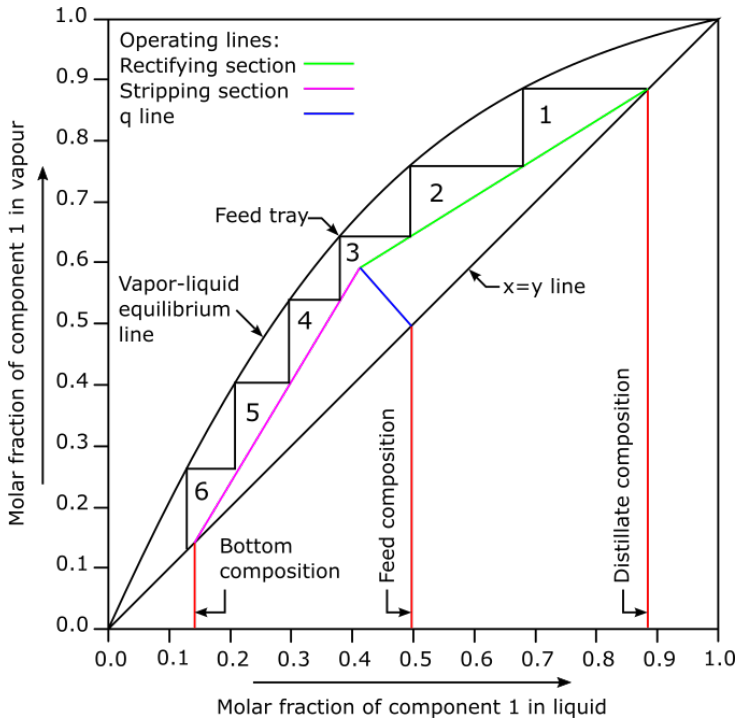


Figure 2.8: Example of the application of the McCabe-Thiele method. Adapted from [10].

We start at the intersection of distillate composition and identity line. The y-coordinate is equal to $y_1^{(N_S)}$. We go horizontally to the vapour-liquid equilibrium line to obtain as x-coordinate $x_1^{(N_S)}$. By going vertically to the operating line of the rectifying section we obtain as y-coordinate $y_1^{(N_S-1)}$. This procedure is repeated until we cross the q-line. Then we continue analogously, but using the operating line of the stripping section, until we meet the bottom product specification. The number of steps drawn into the diagram gives the minimum number of required stages.
From Figure 2.8 and (2.24), we can evaluate the influence of the reflux ratio:
For big values of R the operating line is close to the identity line. The boundary case $R = \infty$, which corresponds to no distillate product ($D = 0$), is called *total reflux*. The operating line equals the identity line and the number of required stages is minimal.
For decreasing R, the slope of the operating line decreases. Since the operating line must intersect the q-line within the equilibrium region, there exists another boundary case where they intersect exactly on the equilibrium line. This is called *minimum reflux*. Here, the number of required stages is infinite, since the above procedure can never reach the q-line.
We will generalize these two cases to multicomponent mixtures in the following two sections.

2.2.2.3 Fenske method

We calculate the minimum number of stages needed to reach given product specifications for a fixed feed under the assumption of constant molar overflow and total reflux conditions. Furthermore, we assume that the mixture can be modelled by constant relative volatilities.
Let LK be the light key and HK the heavy key and let their concentrations in bottom and distillate product be specified. Since the operating lines for both the rectifying and the stripping section equal the identity line, we obtain by using (2.5):

$$\frac{x_{LK}^D}{x_{HK}^D} = \frac{y_{LK}^{(N_S)}}{y_{HK}^{(N_S)}} = \alpha_{LK,HK}\frac{x_{LK}^{(N_S)}}{x_{HK}^{(N_S)}} == \alpha_{LK,HK}\frac{y_{LK}^{(N_S-1)}}{y_{HK}^{(N_S-1)}} = \ldots = \alpha_{LK,HK}^{N_S}\frac{x_{LK}^{(1)}}{x_{HK}^{(1)}} = \alpha_{LK,HK}^{N_S}\frac{x_{LK}^B}{x_{HK}^B}$$

Therefore, the minimum number of stages is obtained as

$$N_S = \frac{\log(\frac{x_{LK}^D}{x_{HK}^D}\frac{x_{HK}^B}{x_{LK}^B})}{\log(\alpha_{LK,HK})}.$$

Note that this calculation only applies in the case of non-perfect separations, else $x_{HK}^D = 0 = x_{LK}^B$.

2.2.2.4 Underwood method

We calculate the minimum reflux ratio needed to reach given product specifications for a fixed feed under the assumption of constant molar overflow and constant relative volatilities.
By $V^{(s)} = RD + qF - B$, this also gives the minimum vapour flow in the stripping section, which is equal to the vapour produced by the reboiler. This is a good measure for the operating costs of a distillation column, since heating is the most cost-intensive process involved.
Note, that we have previously assumed that the components are ordered by decreasing relative volatility $\alpha_1 > \alpha_2 > \ldots > \alpha_{N_C-1} > \alpha_{N_C} = 1$. Further, note that we can assume that there exists

an $\epsilon > 0$, such that $\alpha_c - \alpha_{c+1} > \epsilon$ for all $c = 1, \ldots, N_C - 1$, else the components can not separated by distillation in the constant relative volatility model.

For now, we assume that all components distribute to the distillate and bottom products, this means $x_c^D > 0,\ x_c^B > 0,\ c = 1, \ldots, N_C$. We call the following equation *Underwood equation* for the rectifying section:

$$R + 1 = \sum_{c=1}^{N_C} \frac{\alpha_c x_c^D}{\alpha_c - \varphi} =: f(\varphi). \tag{2.26}$$

We observe

$$f_c(\varphi) := \frac{\alpha_c x_c^D}{\alpha_c - \varphi} \begin{cases} > 0, \varphi < \alpha_c \\ < 0, \varphi > \alpha_c \end{cases}.$$

Furthermore, $f_c(\varphi)$ is continuous and monotonically increasing on both half-planes with a discontinuity at $\varphi = \alpha_c$. Therefore, $f(\varphi)$ is continuous and monotonically increasing on the intervals $]-\infty, \alpha_{N_C}[,]\alpha_{N_C}, \alpha_{N_C-1}[, \ldots,]\alpha_1, \infty[$. Specifically, assuming w.l.o.g $\varphi < \alpha_c + \epsilon/2$:

$$\lim_{\varphi \downarrow \alpha_c} f(\varphi) = \sum_{d=1}^{c-1} \underbrace{\frac{\alpha_d x_d^D}{\alpha_d - \varphi}}_{>0, <2\alpha_1/\epsilon} + \overbrace{\underbrace{\frac{\alpha_c x_c^D}{\alpha_c - \varphi}}_{\uparrow 0}}^{>0, <\alpha_1} + \sum_{d=c+1}^{N_C} \underbrace{\frac{\alpha_d x_d^D}{\alpha_d - \varphi}}_{<0, >-\alpha_1/\epsilon} = -\infty.$$

Analogously, assuming w.l.o.g $\varphi > \alpha_c - \epsilon/2$:

$$\lim_{\varphi \uparrow \alpha_c} f(\varphi) = \sum_{d=1}^{c-1} \underbrace{\frac{\alpha_d x_d^D}{\alpha_d - \varphi}}_{\text{bounded}} + \overbrace{\underbrace{\frac{\alpha_c D x_c^D}{\alpha_c - \varphi}}_{\downarrow 0}}^{>0, <\alpha_1} + \sum_{d=c+1}^{N_C} \underbrace{\frac{\alpha_d x_d^D}{\alpha_d - \varphi}}_{\text{bounded}} = \infty.$$

By the intermediate value theorem, there exists a solution φ_c of (2.26) on each interval $]\alpha_{c+1}, \alpha_c[,\ c = 1, \ldots, N_C - 1$. Furthermore, since $\lim_{\varphi \downarrow -\infty} f(\varphi) = 0$ and $R > 0$, there exists another solution φ_{N_C} on the interval $]-\infty, \alpha_{N_C}[$. Since the solutions of (2.26) are the roots of a polynomial of N_Cth order, there exist no additional solutions.

Analogously, we consider the Underwood equation for the stripping section:

$$S = \sum_{c=1}^{N_C} \frac{-\alpha_c x_c^B}{\alpha_c - \psi} =: \sum_{c=1}^{N_C} g_c(\psi) =: g(\psi). \tag{2.27}$$

By similar argumentation, we derive that (2.27) has solutions ψ_c on the intervals $]\alpha_c, \alpha_{c-1}[,\ c = 2, \ldots, N_C$ and another solution ψ_1 on the interval $]\alpha_1, \infty[$.

Note, that if component c does not appear in the distillate or bottom product, then the corresponding term f_c or g_c vanishes and we do not obtain a solution φ_c on the interval $]\alpha_{c+1}, \alpha_c[$ or ψ_c on the interval $]\alpha_c, \alpha_{c-1}[$, respectively.

For big values of R (and consequently big values of S), the solutions φ_c and ψ_c approach the relative volatility α_c. For decreasing R (and consequently decreasing S), the values φ_c decrease, while the values ψ_c increase. Underwood showed, that at minimum reflux the values φ_c and ψ_{c+1} coincide [11] [12]. We outline the arguments in Appendix A.

The coinciding values, we denote them by ϕ, are solutions of both (2.26) and (2.27) and therefore satisfy the *Underwood feed equation*:

$$\begin{aligned} V^{(r)} - V^{(s)} &= \sum_{c=1}^{N_C} \frac{\alpha_c (D x_c^D + B x_c^B)}{\alpha_c - \phi} \\ \Rightarrow \qquad (1-q) &= \sum_{c=1}^{N_C} \frac{\alpha_c x_c^F}{\alpha_c - \phi}. \end{aligned} \tag{2.28}$$

There are $N_C - 1$ solutions ϕ_c on the intervals $]\alpha_{c+1}, \alpha_c[,\ c = 1, \ldots, N_C - 1$.

We consider a separation with light key component LK and heavy key component HK. Let the distillate product recoveries $r_c := (D x_c^D)/(F x_c^F)$ of the key components be fixed. All components lighter than the light key are recovered completely in the distillate: $r_c = 1$, $c = 1, \ldots, LK - 1$, while all components heavier than the heavy key vanish in the distillate: $r_c = 0$, $c = HK + 1, \ldots, N_C$. The recoveries of the non-key distributing components are unknown. But they are determined by the condition of minimum reflux as follows [13]:

From a previous note, there can only exist coinciding values ϕ_c on $]\alpha_{c+1}, \alpha_c[$ for $c = LK, \ldots, HK-1$. We solve the Underwood feed equation (2.28) for these. Now, we can obtain the values of the $HK-LK$ variables (minimum vapour flow $V^{(r)}$ and unknown recoveries r_c, $c = LK+1, \ldots, HK-1$) by solving the following system of $HK - LK$ linear equations:

$$\frac{V^{(r)}}{F} - \sum_{c=LK+1}^{HK-1} \frac{\alpha_c r_c x_c^F}{\alpha_c - \phi_d} = \sum_{c=1}^{LK-1} \frac{\alpha_c x_c^F}{\alpha_c - \phi_d} + \frac{\alpha_{LK} r_{LK} x_{LK}^F}{\alpha_{LK} - \phi_d} + \frac{\alpha_{HK} r_{HK} x_{HK}^F}{\alpha_{HK} - \phi_d},\ d = LK, \ldots, HK-1 \tag{2.29}$$

where the right hand side is known. Note, that this is simply a reformulation of the Underwood equations for the rectifying section (2.26). The corresponding minimum reflux ratio can be calculated by $R = \frac{V^{(r)}}{D} - 1$ and the corresponding minimum vapour flow in the stripping section by $V^{(s)} = V^{(r)} + (1-q)F$. [14] [15]

Note, that the above calculation also applies to perfect separations. In that case, $r_{LK} = 1$ and $r_{HK} = 0$ which means there are no solutions ψ_{LK} on $]\alpha_{LK}, \alpha_{LK-1}[$ and φ_{HK} on $]\alpha_{HK+1}, \alpha_{HK}[$, but the common solutions remain. Specifically, for a perfect sharp split ($HK = LK + 1$), the procedure reduces to obtaining ϕ_{LK} from (2.28) and calculating $V^{(r)}$ from the single equation (2.29) without additional unknowns.

As an example, we consider the separation of a binary mixture with boiling liquid feed (after [3, p.571-573]). There exists exactly one solution ϕ of the Underwood feed equation (2.28):

$$\begin{aligned} 0 &= \frac{\alpha_1 x_1^F}{\alpha_1 - \phi} + \frac{1 - x_1^F}{1 - \phi} \\ \Leftrightarrow \qquad 0 &= \alpha_1 x_1^F (1-\phi) + (1 - x_1^F)(\alpha_1 - \phi) \\ \Leftrightarrow \qquad 0 &= \alpha_1 x_1^F - \alpha_1 x_1^F \phi + \alpha_1 - \phi - x_1^F \alpha_1 + x_1^F \phi \\ \Leftrightarrow \qquad 0 &= -(\alpha_1 x_1^F + (1 - x_1^F))\phi + \alpha_1 \\ \Leftrightarrow \qquad \phi &= \frac{\alpha_1}{\alpha_1 x_1^F + (1 - x_1^F)}. \end{aligned}$$

Instead of using the reformulation (2.29), we directly insert ϕ in the Underwood equation for the rectifying section (2.26) to obtain:

$$\begin{aligned}
R &= \frac{\alpha_1 x_1^D}{\alpha_1 - \phi} + \frac{1 - x_1^D}{1 - \phi} - 1 \\
&= (\alpha_1 x_1^F + (1 - x_1^F))(\frac{x_1^D}{(\alpha_1 x_1^F + (1 - x_1^F)) - 1} + \frac{1 - x_1^D}{(\alpha_1 x_1^F + (1 - x_1^F)) - \alpha_1}) - 1 \\
&= (\alpha_1 x_1^F + (1 - x_1^F))(\frac{x_1^D}{x_1^F(\alpha_1 - 1)} + \frac{1 - x_1^D}{(1 - x_1^F)(1 - \alpha_1)}) - 1 \\
&= (\alpha_1 x_1^F + (1 - x_1^F))(\frac{x_1^D(1 - x_1^F) - (1 - x_1^D)x_1^F}{x_1^F(1 - x_1^F)(\alpha_1 - 1)}) - 1 \\
&= \frac{(x_1^D - x_1^F)(\alpha_1 x_1^F + (1 - x_1^F)) - (\alpha_1 x_1^F - x_1^F(\alpha_1 x_1^F + (1 - x_1^F)))}{\alpha_1 x_1^F - x_1^F(\alpha_1 x_1^F + (1 - x_1^F))} \\
&= \frac{x_1^D(\alpha_1 x_1^F + (1 - x_1^F)) - \alpha_1 x_1^F}{\alpha_1 x_1^F - x_1^F(\alpha_1 x_1^F + (1 - x_1^F))} \\
&= \frac{x_1^D - \frac{\alpha_1 x_1^F}{\alpha_1 x_1^F + (1 - x_1^F)}}{\frac{\alpha_1 x_1^F}{\alpha_1 x_1^F + (1 - x_1^F)} - x_1^F} \\
&= \frac{x_1^D - y_1^F}{y_1^F - x_1^F}.
\end{aligned}$$

This corresponds to a slope of the operating line of

$$\frac{R}{R+1} = \frac{x_1^D - y_1^F}{y_1^F - x_1^F} \frac{y_1^F - x_1^F}{x_1^D - y_1^F + y_1^F - x_1^F} = \frac{x_1^D - y_1^F}{x_1^D - x_1^F}.$$

This is exactly the slope of the line between the points (x_1^D, x_1^D) and (x_1^F, y_1^F), which are the points of intersection of distillate composition and identity line and q-line and equilibrium curve, respectively. Remember, that the feed is boiling liquid, therefore $q = 1$ and $x^q = x^F$. This shows, that in the binary case the Underwood method reduces to the graphical interpretation given before.

2.2.2.5 Gilliland method

Gilliland established an empirical relation between the number of stages N_S and the reflux ratio R of a distillation column, for known minimum number of stages N_S^{min}(by Fenske method) and minimum reflux ratio R^{min}(by Underwood method). This correlation, pictured in Figure 2.9, can be used to estimate the number of stages N_S for a given value of R (or the other ways round) and select an economic trade-off between the two [3, p. 614].
Note, that the capital cost of a distillation column is driven by its height, which is proportional to number of stages N_S, The operating cost of a distillation column is driven by the steam (and cooling water) cost, which is proportional to the vapour flow in the stripping section of a column $V^{(s)} = RD + qF - B$ and therefore proportional to R.

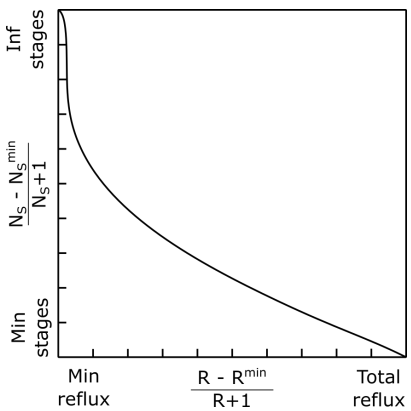


Figure 2.9: Graph of the Gilliland correlation. Adapted from [16].

Chapter 3

Mathematical Optimization

A general mathematical optimization problem can be formulated as follows:

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & x \in A \end{aligned} \tag{OPT}$$

where x are the *variables*, $A \subset \mathbb{R}^n$ is the *feasible set* and $f : A \to \mathbb{R}$ is the *objective*.
In our ansatz discussed in Part III we will encounter different types of optimization problems, which are briefly described in the following. We explain their general form, discuss some note-worthy properties and give a short overview of strategies for solving them numerically.

3.1 Non-linear programming

A prototypical *non-linear program* (NLP) is given by

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & g(x) \leq 0 \\ & h(x) = 0 \\ & x \in B \end{aligned} \tag{NLP}$$

where $B = \prod_{i=1}^{n} [l_i, u_i] \subset \mathbb{R}^n$ is given by the *lower bounds* l_i and *upper bounds* u_i on the variables x_i, $i = 1, \ldots, n$. The *inequality constraints* are given by $g : B \to \mathbb{R}^p$, while the *equality constraints* are given by $h : B \to \mathbb{R}^q$. Now the feasible set is given by $A = \{x \in B : g(x) \leq 0 \wedge h(x) = 0\}$.

3.1.1 Properties

The main result for non-linear programs is the Karush-Kuhn-Tucker theorem. [17]
Let $\bar{x}$ be a local minimum of (NLP), that means there exists a neighbourhood U of $\bar{x}$, such that $f(\bar{x}) \leq f(x)\ \forall x \in U \cap A$. We assume that f, $g_i, i = 1, \ldots, p$ and $h_j, j = 1, \ldots, q$ are continuously differentiable at $\bar{x}$.
Further, we assume that $\bar{x}$ fulfils one of the following two constraint qualifications.

- *linear independence constraint qualification* (LICQ): $\nabla g_i(\bar{x}),\ i \in I(\bar{x}),\ \nabla h_j(\bar{x}),\ j \in \{1, \ldots, q\}$ are linearly independent.

- *Mangasarian-Fromovitz constraint qualification* (MFCQ): $\nabla h_j(\bar{x}),\ j \in \{1,\dots,q\}$ are linearly independent and $\exists d \in \mathbb{R}^n :\ \nabla g_i(\bar{x})^T d < 0,\ i \in I(\bar{x})$ and $\nabla h_j(\bar{x})^T d = 0,\ j \in \{1,\dots,q\}$.

where we define the *set of active constraints* as $I(\bar{x}) := \{i \in \{1,\dots,p\} : g_i(\bar{x}) = 0\}$.
Note that LICQ implies MFCQ and there exist even weaker constraint qualifications that are sufficient for the theorem, but more technical to prove in applications.
Under these assumptions, the theorem states that there exist $\mu \in \mathbb{R}^p, \lambda \in \mathbb{R}^q$ such that

$$\begin{aligned} \nabla f(\bar{x}) + \sum_{i=1}^{p} \mu_i \nabla g_i(\bar{x}) + \sum_{j=1}^{q} \lambda_j \nabla h_j(\bar{x}) &= 0 \\ g_i(\bar{x}) &\le 0, \quad i = 1,\dots,p \\ h_j(\bar{x}) &= 0, \quad j = 1,\dots,q \\ \mu_i &\ge 0, \quad i = 1,\dots,p \\ \mu_i g_i(\bar{x}) &= 0, \quad i = 1,\dots,p \end{aligned} \tag{3.1}$$

The multipliers μ, λ are bounded, if and only if MFCQ holds [18]. They are unique, if and only if LICQ holds [19].

3.1.2 Solution Strategies

One of the oldest approaches to solving NLPs numerically is the *penalty* method, which was first proposed in [20]. The idea is to replace (NLP) by an unconstrained problem of the form

$$\text{minimize} \quad f(x) + \rho P(x) \tag{PNLP}$$

where $\rho > 0$ is the penalty parameter and P is the penalty function, which should fulfil $P(x) = 0\ \forall x \in A$ and $P(x) > 0\ \forall x \notin A$. For example, we could use $P(x) = \sum_{i=1}^{p} (\max(0, g_i(x)))^\alpha + \sum_{j=1}^{q} |h_j(x)|^\alpha$. For ρ larger than a critical value ρ_c, the solution of (PNLP) is the solution of our original problem (NLP). In practice, we solve a series of problems (PNLP) with increasing values of ρ. Each of these is an unconstrained optimization problem that can be solved e.g by Newton's method.
The originally proposed method involved a quadratic penalty function ($\alpha = 2$), but newer methods have been developed for the l_1-penalty function ($\alpha = 1$). A further development of this approach is the *augmented Lagrangian* method, which was first described in [21].
Similarly, we can construct a *barrier* method, by replacing (NLP) with

$$\begin{aligned} \text{minimize} \quad & f(x) + \frac{1}{\rho} B(x) \\ \text{subject to} \quad & x \in A^{\mathrm{o}} \end{aligned}$$

where A^{o} denotes the interior of A, $\rho > 0$ is the barrier parameter and B the barrier function, which should fulfil $B(x) \ge 0\ \forall x \in A^{\mathrm{o}}$ and $B(x) \to \infty\ \forall x \to \partial A$.
This classical approach is out-dated in applications but has been developed further, fuelled by Karmarkar's algorithm for linear programs [22], into the *interior point* (IP) method, the general ideas of which where introduced in [23].

Another approach used in commercial solvers is *sequential quadratic programming* (SQP), which was first proposed in [24]. The idea is to replace the objective in (NLP) by a quadratic approximation and the constraints by a linear approximation at the current solution. The resulting quadratic subproblem can be solved by projection methods or the aforementioned interior point methods. This process is iterated. Note, that in case of no inequality constraints, this approach is equivalent to finding a solution of the KKT-conditions (3.1) with Newton's method.
Details for all the above methods, with results on convergence and detailed discussion of implementation details can be found in [25].

3.2 Mixed-integer non-linear programming

A prototypical *mixed-integer non-linear program* (MINLP) is given by

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & g(x) \leq 0 \\ & h(x) = 0 \\ & x \in B \\ & x_l \in \mathbb{Z}, \quad l = m+1, \ldots, n \end{aligned} \tag{MINLP}$$

where $x_k,\ k = 1, \ldots, m$ are the *continuous variables* and $x_l,\ l = m+1, \ldots, n$ are the *discrete (or integer) variables*. Now the feasible set is given by $A = \{x \in B :\ g(x) \leq 0 \wedge h(x) = 0 \wedge x_l \in \mathbb{Z},\ l = m+1, \ldots, n\}$.

3.2.1 Solution Strategies

Algorithms for solving MINLPs are generally based on two basic concepts:
Firstly, considering *relaxations* to obtain lower bounds on the optimal solution. In general, a relaxation of (OPT) is given by

$$\begin{aligned} \text{minimize} \quad & \tilde{f}(x) \\ \text{subject to} \quad & x \in \tilde{A} \end{aligned} \tag{ROPT}$$

with $\tilde{A} \supset A$ and $\tilde{f}(x) \leq f(x)\ \forall x \in A$.
There are multiple ways of obtaining relaxations for (MINLP):

- Relaxing integrality: We obtain a relaxation by omitting the integrality constraints on some or all of the integer variables.
- Relaxing convex constraints: If g_i is a convex function, we obtain a relaxation by replacing the constraint $g_i(x) \leq 0$ with the set of constraints $g_i(x^{(k)}) + \nabla g_i(x^{(k)})^T(x - x^{(k)}) \leq 0,\ k \in K$ for any points $x^{(k)},\ k \in K$.
- Relaxing non-convex constraints: For some non-linear functions g_i there are known convex under estimators $\tilde{g}_i$, i.e. $\tilde{g}_i(x) \leq g_i(x)\ \forall x \in conv(B)$. Then we obtain a relaxation by replacing the constraint $g_i(x) \leq 0$ with $\tilde{g}_i(x) \leq 0$.

The resulting relaxations are easier to solve, since they are can be treated as NLPs, MILPs or convex MINLPs, respectively.

Secondly, we use *constraint enforcement* to obtain sharper lower bounds. We can tighten a relaxation by adding additional constraints, that exclude the computed solution of the relaxed problem, but no feasible points of the original problem. Further, we can divide the relaxed problem into multiple subproblems, which are again easier to solve. This is called *branching*.

For example, we relax the integrality constraint on variable x_l and compute the optimal solution $\bar{x}$ of the relaxation, which has a fractional value $\bar{x}_l \notin \mathbb{Z}$. Then we can consider the two subproblems obtained from adding either $x_l \leq \lfloor \bar{x}_l \rfloor$ or $x_l \geq \lceil \bar{x}_l \rceil$ as constraints.

Together with upper bounds, which can be obtained from evaluating the objective at any feasible point, we can iteratively narrow down the possible objective value of the optimal solution. Specifically, if we obtain a subproblem with a lower bound that is larger than the current upper bound, we can conclude that this subproblem does not contain the overall optimal solution. Additionally, we can exclude any infeasible subproblems.

These methods are the basis for a lot of algorithms, the most basic of which, that only uses branching on the fractional variables (and excluding problems that are infeasible/not optimal) is called *branch-and-bound* and was first proposed for MINLPs in [26]. Including the generation of additional valid inequalities (also called *cuts*) yields the *branch-and-cut* method, first described in [27].

These approaches generally require the successive solving of a large number of NLPs, that can not be easily hot-started. Therefore other methods, that consider alternating sequences of NLP subproblems and MINLP relaxations have been developed, namely outer approximation [28], generalized Benders decompositon [29] and extended cutting plane method [30].

Note, that for general non-convex problems (non-convex objective or constraint functions) it can not be guaranteed to obtain the global optimum of the NLP subproblems. Therefore they pose more complications. Some methods use piecewise-linear approximation to avoid these problems. Further, there are some generic strategies for obtaining convex relaxations for non-convex problems. Then the method of spatial branching can be used to obtain a convergent algorithm.

More detailed descriptions of all these approaches can be found in [31].

3.3 Mathematical programming with complementarity constraints

A prototypical *mathematical program with complementarity constraints* (MPCC) is given by

$$\begin{aligned} \text{mininimize} \quad & f(x) \\ \text{subject to} \quad & g(x) \leq 0 \\ & h(x) = 0 \\ & x \in B \\ & 0 \leq y \perp z \geq 0 \end{aligned} \tag{MPCC}$$

where $y := (x_1, \ldots, x_m)^T$ and $z := (x_{m+1}, \ldots, x_{2m})^T$ are the *complementing variables*.

3.3.1 Properties

The last constraint has many equivalent reformulations:

$$\begin{aligned} & y^T z = 0 \wedge y \geq 0 \wedge z \geq 0 \\ \Leftrightarrow\ & y_k z_k \leq 0 \wedge y_k \geq 0 \wedge z_k \geq 0, && k = 1, \ldots, m \\ \Leftrightarrow\ & (y_k = 0 \wedge z_k \geq 0) \vee (y_k \geq 0 \wedge z_k = 0), && k = 1, \ldots, m. \end{aligned}$$

Replacing the complementarity constraints by one of the two first expressions gives an NLP.

However, the resulting NLP violates LICQ and MFCQ at all points. Consider the minimal example with two variables $(x_1,\ x_2)$ that are complementing and no additional constraints. Then the reformulation as NLP is given by:

$$\begin{aligned} \text{minimize} \quad & f(x_1, x_2) \\ \text{subject to} \quad & -x_1 \leq 0 \\ & -x_2 \leq 0 \\ & x_1 x_2 \leq 0 \end{aligned}$$

Let $\bar{x}$ be a local minimum, w.l.o.g assume $\bar{x}_1 = 0$ (the case $\bar{x}_2 = 0$ can be treated analogously). Then the gradients of the active constraints $(-1, 0)^T$ and $(x_2, 0)^T$ are linearly dependent. $\lightning$ LICQ

Further, there can not exist $d = (d_1, d_2)^T$: $-d_1 < 0,\ d_1 x_2 < 0$. $\lightning$ MFCQ

This shows, that the multipliers of the KKT-conditions are non-unique and unbounded, which poses problems for a lot of solution strategies.

Therefore, we introduce a new concept: A feasible point $\bar{x}$ of (MPCC) is called *strongly stationary* if and $d = 0$ is the solution of

$$\begin{aligned} \text{minimize} \quad & \nabla f(\bar{x})^T d \\ \text{subject to} \quad & g(\bar{x}) + \nabla g(\bar{x})^T d \leq 0 \\ & h(\bar{x}) + \nabla h(\bar{x})^T d = 0 \\ & d_{y,i} = 0\ , i \in I_y(\bar{x}) \backslash I_z(\bar{x}) \\ & d_{z,i} = 0\ , i \in I_z(\bar{x}) \backslash I_y(\bar{x}) \\ & d_{y,i} \geq 0\ , i \in I_y(\bar{x}) \cap I_z(\bar{x}) \\ & d_{z,i} \geq 0\ , i \in I_y(\bar{x}) \cap I_z(\bar{x}) \end{aligned}$$

with $I_y(\bar{x}) := \{i \in \{1, \ldots, m\} : \bar{y}_i = 0\}$ and $I_z(\bar{x}) := \{i \in \{1, \ldots, m\} : \bar{z}_i = 0\}$, $d_y = (d_1, \ldots, d_m)^T$ and $d_z := (d_{m+1}, \ldots, d_{2m})^T$. Further, we define $I_g(\bar{x}) := \{i \in \{1, \ldots, p\} : g_i(\bar{x}) = 0\}$.

We say *MPEC-LICQ* holds at $\bar{x}$ if $\nabla g_i(\bar{x}), i \in I_g(\bar{x}),\ \nabla h_j(\bar{x}),\ j = 1, \ldots, q,\ \nabla y_i,\ i \in I_y(\bar{x})$ and $\nabla z_i,\ i \in I_z$ are linearly independent. This is a generic property for MPCCs [32].

Now it holds: If $\bar{x}$ is a solution of (MPCC) and MPCC-LICQ holds at $\bar{x}$, then $\bar{x}$ is strongly stationary [33]. Further, by previous remarks the multipliers of the new problem are unique.

3.3.2 Solution Strategies

Applying normal NLP strategies to the NLP reformulation poses difficulties, since the multipliers are non-unique and unbounded. However, some active set strategies have shown promising candidates to overcome these problems [34].
Still, the more promising approach is to relax the complementarity constraints with relaxation parameter $\epsilon > 0$. There are multiple options:

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & g(x) \leq 0 \\ & h(x) = 0 \\ & x \in B \\ & y, z \geq 0 \\ & y_i z_i \leq \epsilon \quad , i = 1, \ldots, m \end{aligned} \tag{Reg(ϵ))}$$

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & g(x) \leq 0 \\ & h(x) = 0 \\ & x \in B \\ & y, z \geq 0 \\ & y^T z \leq \epsilon \end{aligned} \tag{RegComp(ϵ))}$$

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & g(x) \leq 0 \\ & h(x) = 0 \\ & x \in B \\ & y, z \geq 0 \\ & y_i z_i = \epsilon \quad , i = 1, \ldots, m \end{aligned} \tag{RegEq(ϵ))}$$

We solve a series of NLPs with $\epsilon \to 0$. The resulting solutions are generally attracted to strongly stationary points [33].
Alternatively, we can eliminate the complementarity constraint by a penalty approach with penalty parameter ρ:

$$\begin{aligned} \text{minimize} \quad & f(x) + \rho y^T z \\ \text{subject to} \quad & g(x) \leq 0 \\ & h(x) = 0 \\ & x \in B \\ & y, z \geq 0 \end{aligned} \tag{PF(ρ))}$$

Again, we solve a series of NLPs, now with increasing values ρ. In fact, for ρ bigger than a critical value ρ_c, we obtain strongly stationary point [33].
Note, that there is a strong interaction, between the reformulation used and the method employed by the applied NLP solver. The results of multiple benchmark tests in [35] suggest that the penalty formulation with an active set NLP solver is particularly advantageous.

3.4 Generalized disjunctive programming

A prototypical *generalized disjunctive program* (GDP) is given by

$$\begin{aligned}
\text{minimize} \quad & f(x) + \sum_{k \in K} c_k \\
\text{subject to} \quad & g(x) \leq 0 \\
& h(x) = 0 \\
& \bigwedge_{i \in D_k} \begin{bmatrix} Y_{ik} \\ r_{ik}(x) \leq 0 \\ c_k = \gamma_{ik} \end{bmatrix}, \qquad k \in K \\
& \bigvee_{t=1,\ldots,T} \left[\bigwedge_{Y_{jk} \in R_t} Y_{jk} \bigwedge_{Y_{jk} \in Q_t} \neg Y_{jk} \right] = \text{True}
\end{aligned} \tag{GDP}$$

where K is the index set of disjunctions, which are composed of terms with the index set D_k. In each term there is a Boolean variable Y_{ik}. If it is $True$, the inequality $r_{ik}(x) \leq 0$ and the additional cost γ_{ik} are enforced, else they are ignored. Note, that in general, it is assumed that each disjunction is an exclusive-or, which means for each $k \in K$ exactly one variable Y_{ik} is true. The Boolean variables are connected by a logical constraint in conjunctive normal form. For each clause $t = 1, \ldots, T$, R_t is the set of Boolean variables that appear non-negated and Q_t is the set of those that appear negated.

3.4.1 Solution Strategies

An obvious approach to solving GDPs is the reformulation as an MINLP and application of previously discussed solution strategies for MINLPs.
The big-M reformulation of (GDP), first proposed in [36], is given by:

$$\begin{aligned}
\text{minimize} \quad & f(x) + \sum_{k \in K} \sum_{i \in D_k} \gamma_{ik} y_{ik} \\
\text{subject to} \quad & g(x) \leq 0 \\
& h(x) = 0 \\
& r_{ik}(x) \leq M(1 - y_{ik}), \quad k \in K,\ i \in D_k \\
& \sum_{i \in D_k} y_{ik} = 1, \qquad k \in K \\
& Cy \leq c
\end{aligned} \tag{3.2}$$

where $y_{ik} = 1$ corresponds to $Y_{ik} = True$. In that case $r_{ik}(x) \leq 0$ is enforced. Else, the constraint

is redundant, if M is chosen large enough. The remaining constraints represent the disjunctions and the logical constraint.
Alternatively, a convex hull reformulation, introduced in [37], can be used.
However, these reformulation loose the logic structure of the original problem. In order to exploit it, two other approaches have been proposed for convex GDPs: The *branch and bound* method [37] and *logic based outer approximation* [38] translate the identically named methods for MINLPs to GDPs by operating directly on the disjunctions.
A more detailed overview over these methods can be found in [39].

Part II

Review of literature on optimization of distillation sequences

Chapter 4

Problem formulation

The distillation column that has been treated in detail in Chapter 2.2 can facilitate the separation of a binary feed mixtures into (almost) pure components, in case of zeotropic mixtures. Real-world separation tasks, however, often involve multicomponent feed mixtures. In order to obtain (almost) pure components, we need to consider systems of multiple distillation columns, we call them *(distillation) column sequences*.
First, we examine literature, that has treated the problem of describing the set of possible column sequences. We discuss classifications based on the types of separations they facilitate and the number of columns involved. For different classes, we give methods to count the number of distinct sequences. Further, we examine mathematical descriptions of the feasible sequences, that can be used as constraints for the intended optimization.

4.1 Types of distillation configurations

Instead of considering a distillation column sequence with all interconnecting streams, it is easier to first consider the sequence of splits performed by it. We call this a *distillation configuration*.

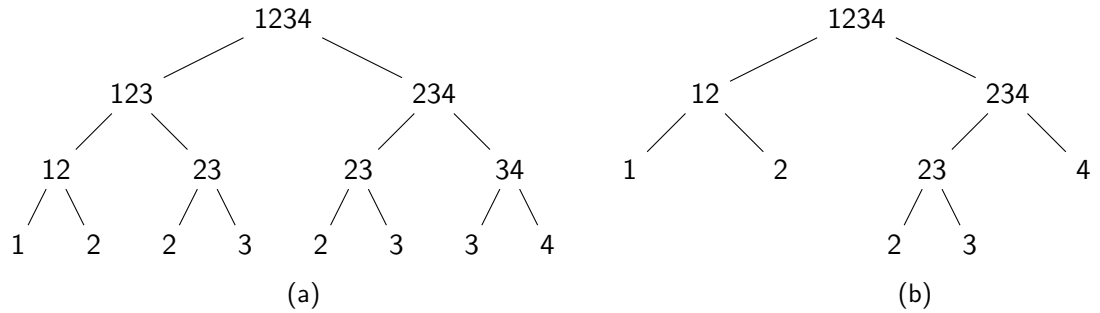


Figure 4.1: Two examples of feasible distillation configurations.

Note, that a split, for example $12|23$, can mean different things, depending on the model used for the actual distillation column. In the rigorous model, with finitely many stages, we will always obtain a non-zero concentration of component 3 in the distillate product. But we can require, that when recovering the (almost) pure components 1 and 2 from the distillate, the impurity caused by it is below a prescribed threshold. In the Underwood model, which assumes infinitely many stages, we can assume that the separation is perfect and require the concentration of component 3 in the distillate

product to be zero. Of course, the concentration of component 1 in the bottom product is treated analogously. In the following it is not necessary to specify which of these models is used, we only use splits to describe our system.

We say a distillation configuration is *feasible* for our separation task, if the splits yield pure components from the feed mixture. We can represent a distillation configuration as a binary tree, where the products of a split are the children of the corresponding feed. Two examples of feasible distillation configurations are given in Figure 4.1.

We can construct a column sequence from a distillation configuration by drawing a column for every split. Further, we will allow to include multiple splits into one distillation column, by considering columns with multiple feeds and side-draws. This can essentially be modelled as multiple (single feed, two product) distillation columns stacked on top of each other, with connecting equations for the product and internal streams. We will do this integration to reach a sequence with the least amount of distillation columns. Note, that this construction does not have to be unique. But we will observe, that for the configurations we focus on uniqueness holds.

4.1.1 Sharp split configurations

One of the earliest approaches to describe feasible distillation configurations was to consider the set of configurations that include only sharp splits. A split is called *sharp*, if the products do not contain any overlapping components [40]. An example of a sharp split separation is given in Figure 4.2.

Note, that we must perform a split between each neighbouring component in order to produce the pure components. By sharpness of the splits, they must be performed exactly once. Therefore, each sharp split configuration involves exactly $N_C - 1$ splits.

Further, we see that no mixture or pure component can occur twice in a sharp split configuration. Therefore, we can not integrate multiple splits into one distillation column. This means, the column sequence corresponding to a sharp split configuration is unique and consists of $N_C - 1$ (single feed, two product) columns. This amounts to $2(N_C - 1)$ column sections, by counting each rectifying and stripping section separately. Note, that this is the minimal number of column sections of a feasible column sequence.

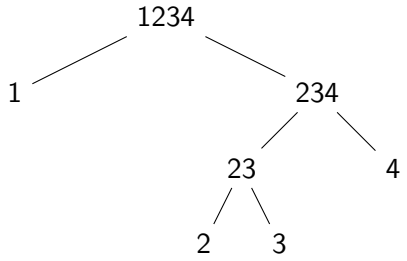


Figure 4.2: Example of a sharp split configuration.

Let S_{N_C} be the number of sharp split configurations that can separate feed mixtures of N_C components. Then $S_1 = 1,\ S_2 = 1$ and for $N_C \geq 3$ we obtain the recurrence relation

$$S_{N_C} = \sum_{c=1}^{N_C-1} S_c S_{N_C-c}. \tag{4.1}$$

The distillate product of the first split may contain the lightest c components, where c must be at least 1 and at most $N_C - 1$, else the column would be redundant. By sharpness of the split, the bottom product then has to contain the heaviest $N_C - c$ components. These product streams act again as feeds for the subsequent columns. Therefore, the number of configurations containing this first split is given as the product of the numbers of configurations that separate feed mixtures of c and $N_C - c$ components. The sum over all the first splits gives the total number of configurations.

These numbers are connected to the *Catalan numbers* C_i by the relation $C_i = S_{i+1},\ i \in \mathbb{N}$. In Appendix B, we give a detailed proof of the following closed-form expression:

$$C_i = \frac{1}{i+1}\binom{2i}{i} = \frac{(2i)!}{i!(i+1)!}$$

which implies

$$S_{N_C} = \frac{1}{N_C}\binom{2N_C-2}{N_C-1} = \frac{(2N_C-2)!}{(N_C-1)!N_C!}.$$

The values of S_{N_C} for $2 \leq N_C \leq 8$ are shown in Table 4.1.

N_C	2	3	4	5	6	7	8
S_{N_C}	1	2	5	14	42	132	429

Table 4.1: Number of sharp split configurations S_{N_C} separating N_C components, for $2 \leq N_C \leq 8$.

4.1.2 Exhaustive configurations

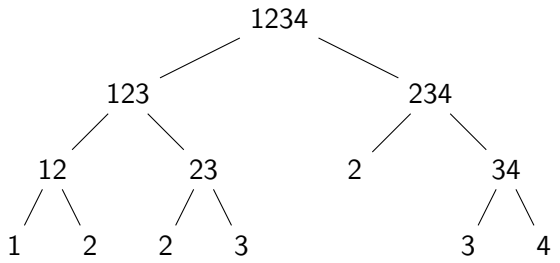


Figure 4.3: Example of an exhaustive configuration.

We call feasible distillation configurations that involve any (sharp and non-sharp) splits *exhaustive configurations* and define E_{N_C} as the number of exhaustive configurations that can separate feed mixtures N_C components. Then $E_1 = 1,\ E_2 = 1$ and for $N_C \geq 3$ we obtain the recurrence relation

$$E_{N_C} = \sum_{c=1}^{N_C-1} \sum_{d=N_C-c}^{N_C-1} E_c E_d.$$

The distillate product of the first split may contain the lightest c components, where c must be at least 1 and at most $N_C - 1$, else the column would be redundant. The bottom product may contain the heaviest d components, where d must be at least $N_C - c$, since no component can vanish, and at most $N_C - 1$.

There exists no (known) closed-form representation for this sequence of numbers, but we can calculate the values E_{N_C} for $2 \leq N_C \leq 8$ by using the recursion formula. The results are given in Table 4.2.

N_C	2	3	4	5	6	7	8
E_{N_C}	1	3	22	719	556,456	310,474,067,253	96,394,492,430,473,282,930,992

Table 4.2: Number of exhaustive configurations E_{N_C} separating N_C components, for $2 \leq N_C \leq 8$.

Unlike in the case of sharp split configurations, for exhaustive configurations there can exist multiple distinct column sequences corresponding to the same configuration. For example, consider the exhaustive configuration pictured in Figure 4.3. There are two distinct corresponding column sequences, both involving 4 columns, given in Figure 4.4.

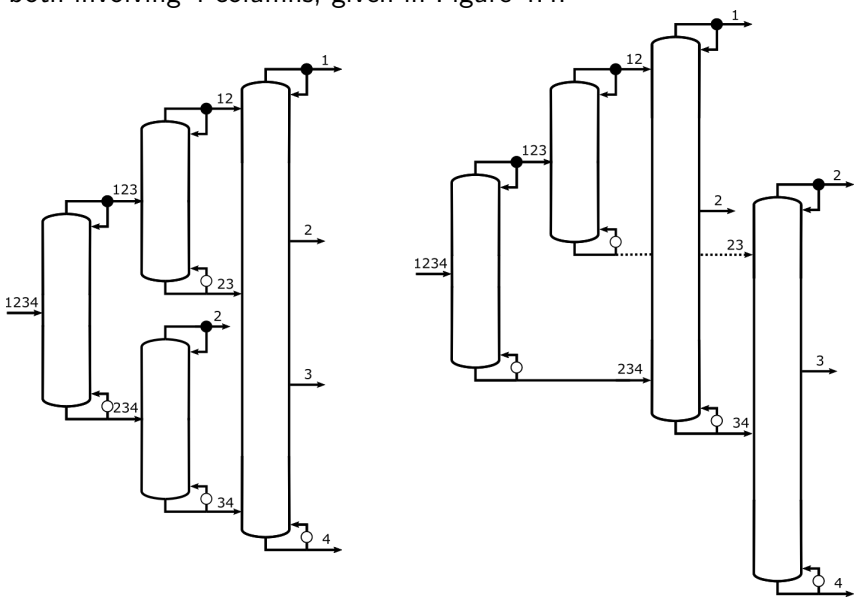


Figure 4.4: Two distinct column sequences corresponding to the configuration in Figure 4.3.

In [41] the authors named the number of distinct column sequences corresponding to a configuration *multiplicity*. They derived an analytic expression for it and calculated the number of exhaustive configurations with multiplicity EM_{N_C}, i.e. the number of distinct column sequences corresponding to exhaustive configurations separating N_C components, for $2 \leq N_C \leq 6$. The results are given in Table 4.3.

N_C	2	3	4	5	6
EM_{N_C}	1	3	24	1,352	4,074,863

Table 4.3: Number of exhaustive configurations with multiplicity EM_{N_C} separating N_C components, for $2 \leq N_C \leq 6$. Data from [41].

4.1.3 Basic configurations

We say an exhaustive configuration is a *basic configuration*, if there exists a corresponding column sequence, that involves exactly $N_C - 1$ distillation columns and recovers each component in exactly one product stream [42].

Note, that by previous remarks all sharp split configurations are basic configurations. But there are basic configurations that include non-sharp splits. Consider, for example, the distillation configuration given in Figure 4.1b. The first split is non-sharp, nevertheless there exists a corresponding column sequence with $N_C - 1 = 3$ columns, given in Figure 4.5.

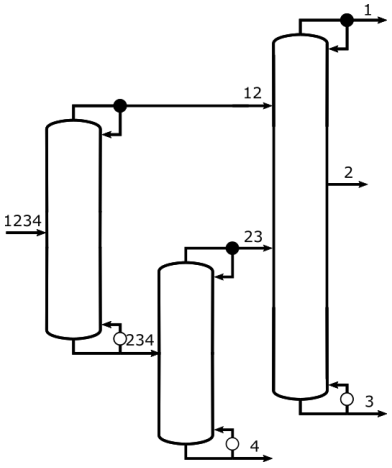


Figure 4.5: Column sequence corresponding to the non-sharp basic configuration in Figure 4.1b.

Further, there exist exhaustive configurations that are non-basic. Consider the distillation configuration given in Figure 4.3 and its corresponding column sequences in Figure 4.4. Note, that both sequences involve $N_C = 4$ columns. Further, since the pure component 2 is recovered as a distillate product in the two splits $2|3$ and $2|34$, there is no possible integration that leads to $N_C - 1$ columns. Let B_{N_C} be the number of basic configurations that can separate feed mixtures of N_C components. Based on the approach in Section 4.2.2 we can construct an efficient method to obtain the value B_{N_C} for any N_C, which we discuss in more detail in Section 6.2.1.1. The values of B_{N_C} for $2 \leq N_C \leq 8$ are displayed in Table 4.4.

N_C	2	3	4	5	6	7	8
B_{N_C}	1	3	18	203	4,373	185,421	15,767,207

Table 4.4: Number of basic configurations B_{N_C} separating N_C components, for $2 \leq N_C \leq 8$.

The analytic expression for the multiplicity evaluates to 1 for every basic configuration [41]. Therefore the column sequence corresponding to a basic configuration is unique.

4.1.4 Comparison

In [41] the authors compared the 24 column sequences, corresponding to exhaustive configurations, that can separate feed mixtures of $N_C = 4$ components. They used the Underwood method to calculate the corresponding minimum vapour flows, under the assumption of boiling liquid feed, vapour distillate and liquid bottom interconnecting streams and liquid products. As mentioned previously, this is a good measure for the operating costs of a column sequence.
They considered 15 different feed compositions by iterating through the power set of the set of components. For each element, they fixed the concentrations of the contained components to 5%, while the concentrations of the remaining components are set to an equal value, such that the summation equation is fulfilled. Note, that we must exclude the set containing all components, since the corresponding composition does not satisfy the summation equation.
Further, they considered eight different separation settings by fixing the three relative volatilities between neighbouring components either to 1.1 or 2.5.
For the resulting 120 test scenarios, they calculated the minimum vapour flow for each of the 24 column sequences. We discuss the results of the two comparisons they performed: "the best sharp split configuration vs. the best non-sharp basic configuration" and "the best basic configuration vs. the best non-basic exhaustive configuration".
Firstly, they showed that in general (except for one case) the sharp split configurations performed worse than the (non-sharp) basic configurations, in some cases by more than 50%. This motivates that non-sharp split configurations should be included in the search space formulation. In some cases though, the sharp split configurations performed almost as well as the (non-sharp) basic configurations. Since the former, in general, have less column sections, therefore less investment costs, and less transfer streams, therefore better operationality, sharp split configurations should not be eliminated from the search space.

Secondly, they discovered that in all cases the non-basic exhaustive configurations performed worse than the basic configurations. Since the former also have a higher (potentially much higher) number of columns and worse operationality, it is a suitable restriction to only consider basic configurations in the search space. A further advantage of this restriction is, that the number of corresponding column sequences grows much more slowly, as can be seen by comparing Table 4.4 and Table 4.3. Lastly, for basic configurations, unlike for non-basic exhaustive configurations, there is a one-to-one correspondence between distillation configuration and column sequence.
Though these results may not completely generalize to models other than the very simplified Underwood method, we take them as a strong indication to choose the set of basic configurations as a reasonable search space.

4.2 Search space formulations

In this section we discuss different approaches to modelling the set of basic configurations mathematically, focussing on their advantages and disadvantages in terms of complexity and usability.

4.2.1 Graph approach

One of the earliest approaches to describe the sets of basic configurations systematically, the *graph approach*, was formalized recently in [43].
Consider a graph with nodes for all possible combinations of components, also called *submixtures*. Since there can only exist combinations of consecutive components, there are $N_C - c + 1$ submixtures with c components. Therefore we obtain the number of nodes:

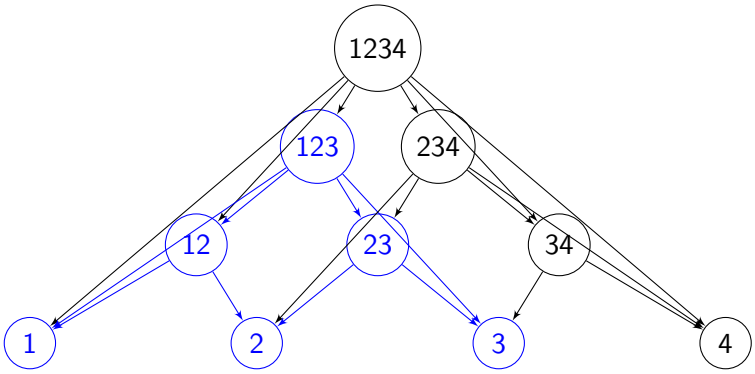


Figure 4.6: Graph corresponding to a four component feed mixture. Subgraph (blue) corresponding to a three component feed mixture.

$$\sum_{c=1}^{N_C}(N_C - c + 1) = \sum_{c=1}^{N_C} c = \frac{N_C(N_C+1)}{2}.$$

Further, we introduce directed edges pointing from a submixture to its possible products. For a combination of c components, $c \geq 2$, there are $(c-1)$ possible distillate products (most volatile only to all but least volatile) and $(c-1)$ possible bottom products (least volatile only to all but most volatile). Therefore we obtain the number of edges:

$$\begin{aligned} 2\sum_{c=2}^{N_C}(N_C - c + 1)(c-1) &= 2\sum_{c=1}^{N_C-1}(N_C - c)c \\ &= 2\left[N_C\frac{(N_C-1)N_C}{2} - \frac{(N_C-1)N_C(2N_C-1)}{6}\right] = \frac{(N_C+1)(N_C-1)N_C}{3}. \end{aligned}$$

The resulting graph for a four component feed mixture is given in Figure 4.6. Note, that we obtain the graph for a three component feed mixture as a subgraph, marked blue in the same figure.

Let N be the set of nodes and $E \subset N \times N$ be the set of edges of the directed graph $G = (N, E)$. Now every distillation configuration can be represented as a subgraph of G, similar to the binary tree representation, but with each submixture projected onto one node. A basic configuration is described by the following requirements:

(G1) If a node is either the feed node or a non-pure-component node with an incoming edge, then it must have exactly one outgoing distillate edge and exactly one outgoing bottom edge. ("Ensure mass balance")

(G2) There must be at least one incoming edge for all pure-component nodes. ("Ensure that pure components are made")

(G3) If a node is either the feed node or a non-pure-component node with an incoming edge, then the number of components of that node must be less or equal the sum of the numbers of components of the successor nodes. ("Ensure that no components vanish")

(G4) A node can not have more than one incoming distillate edge or more than one incoming bottom edge. ("Ensure basicity")

The last requirement ensures that there exists a corresponding column sequence with $N_C - 1$ columns, since we can integrate multiple splits into one column as long as the common products are produced once as distillate and once as bottom.
We introduce the binary variables $X(e) \in \mathbb{B}, e \in E$. The corresponding distillation configuration is given by the set of all edges e with $X(e) = 1$ and their incident nodes. Then the above requirements can be expressed as equations and inequalities for X, for example for a four component feed mixture:

$$
\begin{aligned}
X(1234,123) + X(1234,12) + X(1234,1) &= 1 \\
X(1234,234) + X(1234,34) + X(1234,4) &= 1 \\
X(123,12) + X(123,1) &= X(1234,123) \\
X(123,23) + X(123,3) &= X(1234,123) \\
X(234,23) + X(234,2) &= X(1234,234) \\
X(234,34) + X(234,4) &= X(1234,234) \\
X(12,1) &= X(1234,12) + X(123,12) \\
X(12,2) &= X(12,1) \\
X(23,2) &\geq X(123,23) \\
X(23,2) &\geq X(234,23) \\
X(23,2) &\leq X(123,23) + X(234,23) \\
X(23,3) &= X(23,2) \\
X(34,3) &= X(1234,34) + X(234,34) \\
X(34,4) &= X(34,3)
\end{aligned}
\qquad \text{(G1)}
$$

Note the special form of the constraints for $X(23,2)$. It is feasible that both $X(123,23)$ and $X(234,23)$ are non-zero, which models the case that 23 is produced as a side-draw. Therefore we

can not use an equality, but rather have to ensure by inequalities that there are outgoing edges if and only if (at least) one of the two is non-zero.

$$\begin{aligned} X(1234,1)+X(123,1)+X(12,1) &\geq 1 \\ X(234,2)+X(23,2)+X(12,2) &\geq 1 \\ X(123,3)+X(23,3)+X(34,3) &\geq 1 \\ X(1234,4)+X(234,4)+X(34,4) &\geq 1 \end{aligned} \tag{G2}$$

$$\begin{aligned} X(1234,1)+X(1234,4)+2\ X(1234,12)+2\ X(1234,34)+\ldots& \\ \ldots 3\ X(1234,123)+3\ X(1234,234) &\geq 4 \\ X(123,1)+X(123,3)+2\ X(123,12)+2\ X(123,23) &\geq 3\ X(1234,123) \\ X(234,2)+X(234,4)+2\ X(234,23)+2\ X(234,34) &\geq 3\ X(1234,234) \end{aligned} \tag{G3}$$

Note, that analogous constraints for two component submixtures are redundant with the mass balances. For example $X(12,1)+X(12,2) \geq 2\ X(1234,12)$ and $X(12,1)+X(12,2) \geq 2\ X(123,12)$ must be fulfilled, since $X(1234,12) = 1$ or $X(123,12) = 1$ directly implies $X(12,1) = 1$ and $X(12,2) = 1$ by (G1).

$$\begin{aligned} X(234,2)+X(23,2) &\leq 1 \\ X(123,3)+X(23,3) &\leq 1 \end{aligned} \tag{G4}$$

Note, that analogous equations for the remaining submixtures that could be produced twice as distillate/bottom ($1,\ 4,\ 12$ and 34) are redundant with the mass balances. For example $X(1234,1)+X(123,1)+X(12,1) \leq 1$ is fulfilled with equality, since $X(1234,1)+X(123,1)+X(12,1) = X(1234,1)+[X(1234,123)-X(123,12)]+[X(1234,12)+X(123,12)] = 1$ by (G1). Note, that this also means that the product constraints for the components 1 and 4 are redundant.
We obtain a system of linear equations and inequalities for the $\mathcal{O}(N_C^3)$ binary variables $X(e)$, $e \in E$. It is not straightforward how to generalize the above constraints to arbitrary values of N_C. In fact, all published graph-based approaches rely on manually listing all possible splits and streams [41].

4.2.2 Matrix approach

Efforts to generalize the graph approach resulted in the *matrix approach*, which was introduced in [44].
In terms of the graph model, the idea of this approach is to characterize a configuration by the involved nodes, instead of the involved edges.
We can construct a one-to-one correspondence between the submixtures of a N_C-component feed and the entries of an upper diagonal $N_C \times N_C$ matrix: The entry at (i,j) corresponds to the submixture $i \cdots N_C - j + i$. Note, that its most volatile component is given by the row index i and its number of components is given by $N_C - j + 1$, which decreases with the column index j.

For example, for a four component feed mixture:

$$\begin{pmatrix} 1234 & 123 & 12 & 1 \\ & 234 & 23 & 2 \\ & & 34 & 3 \\ & & & 4 \end{pmatrix}$$

Note, that a submixture can only produce submixtures as distillate, that have the same most volatile component. Similarly, it can only produce submixtures as bottom, that have the same least volatile component. This means, possible distillate products of the entry (i, j) are on the same row to the right, given by the entries $(i, j+m)$, $m = 1, \ldots, N_C - j$. Similarly, possible bottom products are on the same diagonal to the right and down, given by the entries $(i+m, j+m)$, $m = 1, \ldots, N_C - j$.

We introduce the binary variables $X(i, j) \in \mathbb{B}, j = 1, \ldots, N_C, i = 1, \ldots, j$. The corresponding distillation configuration involves exactly the submixtures corresponding to matrix entries (i, j) with $X(i, j) = 1$. Since the feed mixture and all pure components must be present, we fix $X(1, 1) = 1$ and $X(i, N_C) = 1, i = 1, \ldots, N_C$.

Now, the distillation configuration corresponding to X is found as follows:

We start from the upper left corner where $X(1, 1) = 1$ and find the first non-zero entries on the same row to the right, and on the same diagonal to the right and down. The submixture corresponding to the former is the distillate product, while the one corresponding to the latter is the bottom product of the feed split. We continue this procedure iteratively, using the obtained entries as new starting points, until we reach the last column, whose entries correspond to the pure components.

Consider for example the matrix below:

$$X = \begin{pmatrix} 1 & 0 & 1 & 1 \\ & 1 & 1 & 1 \\ & & 0 & 1 \\ & & & 1 \end{pmatrix} \tag{4.2}$$

The corresponding distillation configuration is shown in Figure 4.7:

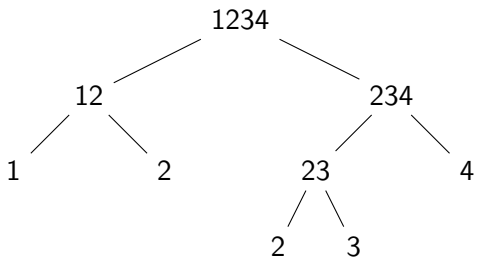


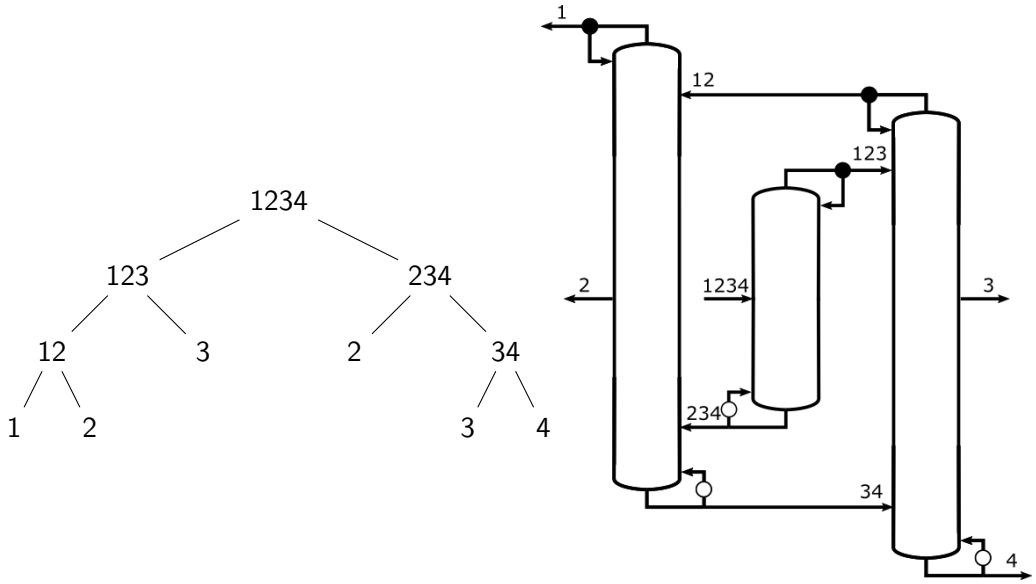
Figure 4.7: Distillation configuration corresponding to the matrix X in (4.2).

By integrating the splits $1|2$ and $2|3$ in one column, we can obtain a corresponding column sequence involving $N_C - 1 = 3$. This means, the distillation configuration in Figure 4.7 is a basic configuration.

Note, that there might be lines crossing each other:

$$X = \begin{pmatrix} 1 & 1 & 1 & 1 \\ & 1 & 0 & 1 \\ & & 1 & 1 \\ & & & 1 \end{pmatrix} \tag{4.3}$$

This matrix corresponds to the so called *satellite configuration*. Its binary tree representation and corresponding column sequence involving $N_C - 1 = 3$ distillation columns are shown in Figure 4.8.



(a) Binary tree representation (b) Integrated column sequence

Figure 4.8: Satellite configuration, corresponding to the matrix X in (4.3).

But most importantly, note, that by construction there can never occur parallel lines. Therefore, no submixture can be produced twice as distillate or twice as bottom, which means we do not obtain non-basic configurations by construction.
We have to ensure that we obtain feasible distillation configurations, by imposing two demands, similarly to those in the graph approach:

(M1) Every submixture, except the feed mixture, that is involved in the configuration, must be produced as a (distillate or bottom) product of an involved submixture ("Ensure mass balance" and "Ensure that pure components are made")

(M2) The number of components of every involved submixture, except the pure components, must be less or equal the sum of the numbers of components of its products ("Ensure that no components vanish")

These demands can be expressed in terms of the variables $X(i,j)$ as follows:

(M1) For all submixtures, except the feed mixture, ($\forall j = 2,\dots,N_C, i = 1,\dots,j$) that are involved in the configuration (such that $X(i,j) = 1$), there must exist a submixture that can produce it as a distillate ($\exists m \in \{1,\dots,j-1\} : X(i,j-m) = 1$) or bottom ($\exists m \in \{1,\dots,i-1\} : X(i-m,j-m) = 1$). This is equivalent to the following set of inequalities:

$$\forall j = 2,\dots,N_C, i = 1,\dots,j : \sum_{m=1}^{j-i} X(i,j-m) + \sum_{m=1}^{i-1} X(i-m,j-m) \geq X(i,j) \qquad \text{(M1)}$$

(M2) For all submixtures, except the pure components, ($\forall j = 1,\dots,N_C-1, i = 1,\dots,j$) that are involved in the configuration (such that $X(i,j) = 1$), the number of components ($N_C - j + 1$) must be less or equal to the sum of the numbers of components of the distillate and bottom products.

Let $M := \{1,\dots,N_C - j\}$. Then the distillate product corresponds to the entry $(i, j + m_d)$ with $m_d = \min\{m : m \in M \wedge X(i,j+m) = 1\}$. Its number of components is given by $N_C-(j+m_d)+1$. Similarly, the bottom product corresponds to the entry $(i + m_b, j + m_b)$ with $m_b = \min\{m : m \in M \wedge X(i + m_b, j + m_b) = 1\}$ and its number of components is given by $N_C - (j + m_b) + 1$.

Since $N_C-(j+m_d)+1+N_C-(j+m_b)+1 \geq N_C-j+1$ is equivalent to $m_d+m_b+j \leq N_C+1$, we obtain the following set of inequalities:

$$\forall j = 1,\dots,N_C-2, i = 1,\dots,j, X(i,j) = 1 : \min_{\substack{m_d \in M \wedge \\ X(i,j+m_d)=1}} m_d + \min_{\substack{m_b \in M \wedge \\ X(i+m_b,j+m_b)=1}} m_b + j \leq N_C + 1 \qquad \text{(M2)}$$

Note, that we can exclude the case $j = N_C - 1$ (two component submixture), since its products must be the pure components and therefore the above inequality is trivial.

We obtain a formulation with the following number of free variables:

$$\sum_{j=2}^{N_C-1} j = \frac{(N_C-1)N_C}{2} - 1 = \frac{(N_C+1)(N_C-2)}{2} = \mathcal{O}(N_C^2)$$

and number of constraints:

$$\sum_{j=2}^{N_C} j + \sum_{j=1}^{N_C-2} j = 2(\sum_{j=1}^{N_C} j) - 1 - (N_C-1) - N_C = N_C(N_C-1) = \mathcal{O}(N_C^2).$$

These equations are completely general in N_C, which is an advantage compared to the graph approach. Further, the number of variables is reduced by one order of magnitude. Therefore this formulation can be used as a basis to enumerate all basic configurations efficiently, see Section 6.2.1.1.

However, the constraints (M2) are not only non-linear, but non-smooth, so they are very difficult to handle as constraints for an optimization problem. Therefore, we derive a hybrid method in Section 6.2.1.2, which gives more suitable constraints.

Since the Underwood shortcut method only operates on splits, it can be used as a model for the distillation columns for both methods, as we demonstrate in Section 6.2.2. Introducing rigorous models, however, is not straightforward and will not be pursued here.

4.2.3 Superstructure approach

Another approach to describe the set of feasible basic configurations is the use of a superstructure. This idea dates back to [45].

We combine the elements of all column sequences corresponding to basic configurations to create a superstructure consisting of distillation columns and interconnecting streams. Each of the basic configurations, more precisely its corresponding column sequence, is obtained by setting certain streams within the superstructure to zero. As an example, the superstructure for a four component feed mixture is shown in Figure 4.9.

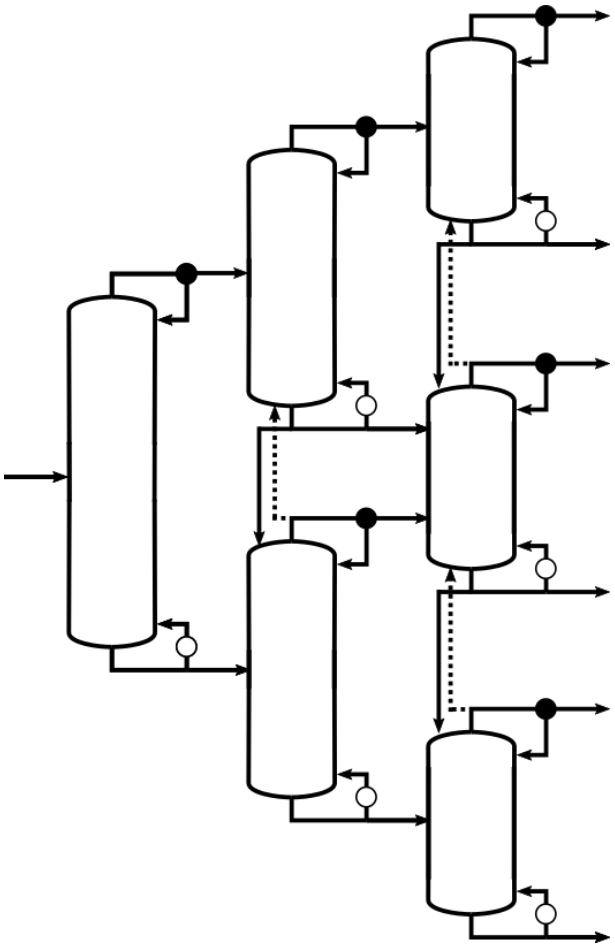


Figure 4.9: Superstructure for a four component feed mixture.

Since this formulation already includes distillation columns instead of the more abstract concept of splits, it is easiest to combine with rigorous models of a distillation column, as we demonstrate in Sections 6.1.1.1 and 6.1.1.2. We will give a detailed derivation of the constraints on the interconnecting streams that restrict the feasible set to the column sequences corresponding to basic configurations there, see especially Section 6.1.2.

However, this approach certainly has its disadvantages: Both the layout of the superstructure and its constraints are difficult to generalize to other values of N_C. Further, for $N_C = 4$ and higher, there exist column sequences corresponding to basic configurations that can not be represented in the superstructure. The first and most prominent example is the satellite configuration, for its corresponding column sequence see Figure 4.8b.

Chapter 5

Solution strategies

Before we go into detail on the approaches motivated in the last chapter, we briefly discuss some additional solution strategies that have been developed for the optimization of distillation columns. We highlight corresponding results that are relevant for our following examinations.

5.1 Sequential optimization algorithm

One of the oldest and most obvious approaches for finding the optimal operating conditions for a given distillation sequence, is to sequentially calculate the optimal operating conditions (and intermediate product compositions) for each distillation column, starting from the feed mixture. This approach was formalized and examined in [46], using the Underwood shortcut method to calculate the minimum vapour flow and corresponding optimal intermediate product compositions in each distillation column. This yields a purely analytical procedure, without the need to solve an optimization problem. Therefore it is computationally very cheap and stable. Combining this approach with a total enumeration of all basic configurations, yields a rank-list of them, based on their minimum vapour flow. This can be used to find the optimal distillation sequence.
However, the case studies in [46] show, that for some basic configurations this sequential procedure does not obtain the overall minimum vapour flow, which was calculated by simultaneously minimizing the vapour flow in all distillation columns using the same Underwood model as constraints. These results show, that it is necessary to treat distillation column sequences as whole and not reduce them to the subproblem of one distillation column.

5.2 Optimization based on the graph approach

In [43] they use the constraints developed using the graph approach, see Section 4.2.1, to describe the set of basic configurations. They couple them with equations derived from the Underwood method to obtain an MINLP. Then they compare two solution strategies:

- They enumerate all basic configurations, by iterating through all possible settings of the binary variables describing the subgraph, and solve the resulting NLP to find the corresponding minimum vapour flow. The optimal distillation sequence is obtained from the resulting rank-list.

- They solve the MINLP to calculate the optimal distillation sequence and its corresponding minimum vapour flow.

They observed that solving the MINLP is computationally expensive, very sensitive to initial guesses and sometimes fails to obtain a feasible solution or obtains a suboptimal one. Solving a series of NLPs, however, is more efficient (and bears the potential for parallelization) and stable.
However, for larger problems $N_C > 5$, they recognize the exponential growth of the number of basic configurations and the resulting computational effort needed for a total enumeration.
These results act as a comparison case for the matrix-based approaches, that we develop and examine in Part III. Specifically, we hope that the novel hybrid method performs better than the graph-based method.

5.3 Optimization based on the State-Task-Network approach

An alternative to the previously discussed search space formulations is the State-Task-Network formulation introduced in [47]. Every possible submixture corresponds to a state and every possible split to a task. Obviously, there are only certain tasks that can be executed on each state. In [48] they simplify the resulting formulation to obtain a formulation where each split is treated like a pseudo-column. If the split is not involved in the configurations, the pseudo-column acts as a bypass, which is modelled using a GDP.
Again, the Underwood shortcut method is used to evaluate the minimum vapour flow corresponding to a distillation column sequence.
They further extend the approach to thermally coupled distillation column sequences, where all or some of the reboilers and condensers are replaced by (backwards) interconnecting streams between the columns. This is a useful extension for industrial applications.
However, the highly specialized GDP formulation limits the applicability to most commercial solvers. Additional reformulations have to be introduce which increase the complexity of the problem.
These results act as a comparison case for the superstructure approaches, that we develop and examine in Part III.

5.4 Genetic algorithm

A completely different approach to the problem was chosen in [49]. They choose to avoid the numerical problems arising from combining the structural decisions with rigorous modelling of the distillation, by proposing an unconstrained optimization problem that can be solved using a problem specific genetic algorithm.
A solution is encoded by its distillation column sequence, operating conditions in the columns and product compositions. They introduce a penalty for not attained product specifications and another for infeasibility of the operating conditions. The latter is derived by considering the distillation profiles of the stripping and rectifying sections for given specifications on the operating conditions and product compositions. The specifications are feasible, if the profiles intersect. The minimal distance between the profiles is taken as a measure of infeasibility. This approach is made computationally efficient

by using orthogonal collocation methods [50]. The resulting method can also be applied to assess azeotropic mixtures.

We obtain an unconstrained optimization problem with a the generalized fitness function, consisting of the original objective and the introduced penalty terms, as objective.

This is solved by using an evolutionary approach: An initial population is created and its fitness evaluated. The best members are selected, recombined and mutated to yield a new population. This procedure is iterated for a fixed number of generations. A schematic representation of the resulting problem specific genetic algorithm is given in Figure 5.1.

However, because of the nature of the genetic algorithm, there is no optimality measure for the obtained solution. Further, the performance of the algorithm is highly dependent on the chosen parameters (crossover/mutation probabilities and weighting of cost/penalty). Moreover, the set of feasible distillation column sequences and their possibilities for recombination and mutation have to be evaluated anew for each value of N_C.

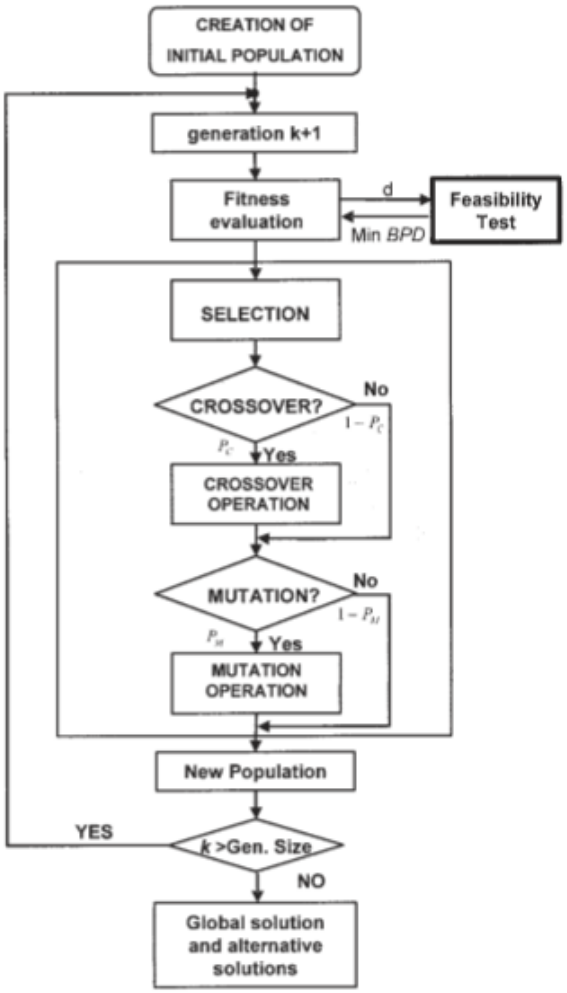


Figure 5.1: Schematic representation of the genetic algorithm. Reproduced from [49].

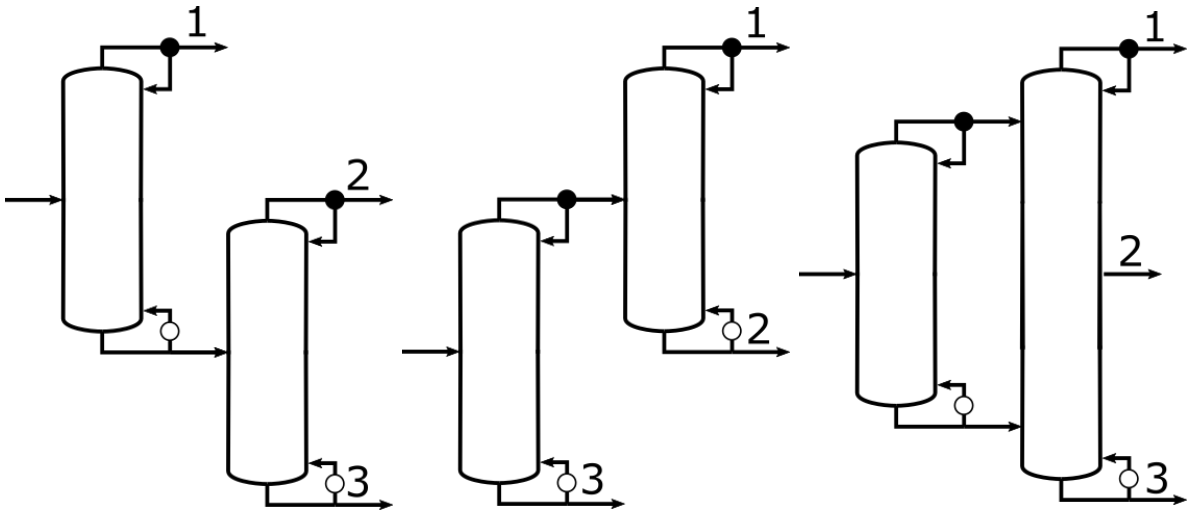
Part III

New approaches for optimization of distillation sequences

Chapter 6

Problem formulation

As hinted in the introduction and motivated in Part II, in this part we want to develop and examine two new approaches for finding the optimal distillation column sequence for a given separation task: Firstly, we try to introduce rigorous models of distillation columns into the problem formulation, which has not been done (outside of the Genetic algorithm) in the reviewed literature. Since these models are based on fewer simplifications than the widely used Underwood model, they bear the potential of yielding more accurate and detailed information. As motivated before, to describe the set of basic configurations in this approach, the superstructure formulation is most suitable.
Secondly, we try to improve previous approaches using matrix-based formulations by deriving a hybrid formulation for describing the set of basic configurations. As motivated before, we combine this formulation with the Underwood model for distillation columns.
For examining the potential of both approaches, we consider the simplest possible scenario, a three component feed mixture ($N_C = 3$). Then there exist three basic configurations, their corresponding column sequences are shown in Figure 6.1.



(a) Direct sequence. (b) Indirect sequence. (c) Prefractionator.

Figure 6.1: Column sequences corresponding to the three basic configurations for separating a three component feed mixture.

We consider a feed of given mass flow F^{all} and composition x^F. We assume that the feed, all connecting streams between columns and the product streams are liquid at their boiling point. We want to find the optimal column sequence and corresponding operating conditions to separate the feed mixture into (almost) pure components.

In order to be able to compare both approaches, we need to formulate an objective that can be evaluated in both models. Therefore, we define the optimal distillation column sequence, as the one that accomplishes the separation task at minimal vapour flow, which is calculated as the sum of the vapour flows at the bottoms of both columns. As discussed previously, this is a good measure for the reboiler duty and therefore operating costs of a column sequence.

6.1 Superstructure approach

In this section, we derive in detail the problem formulation for finding the optimal distillation column sequence using the superstructure approach and the rigorous model of a distillation column.

Since the rigorous model is not suitable for some of the solution strategies examined in Section 8.2, we additionally derive an analogous formulation using the simplified model of a distillation column that arises from the assumption of constant molar overflow and constant relative volatilities.

We consider the superstructure pictured in Figure 6.2, which also introduces all variables. They are named similarly to Section 2.2.

We choose this very general formulation to simplify the notation for the MESH equations. Note, that it allows to distribute the feed and connecting streams to multiple stages (or potentially optimize for the optimal entry stage). Similarly, it allows side-draws from multiple stages. However, in the following we will consider the corresponding stages as (fixed) parameters n_F,n_{CC},n_{CR} and n_{PS}:

$$\begin{aligned} F^{(n_F)} &= F^{all} \\ F^{(n)} &= 0, \quad n \neq n_F \\ CC^{(n)} &= 0, \quad n \neq n_{CC} \\ CR^{(n)} &= 0, \quad n \neq n_{CR} \\ PS^{(n)} &= 0, \quad n \neq n_{PS}. \end{aligned}$$

We will assume $n_{CC} \geq n_{PS} \geq n_{CR}$, which is the only sensible ordering.

Additional parameters, that need to be fixed (but may be changed between different runs of optimizations) are the number of stages in both columns, here denoted by $N_S[1], N_S[2]$.

We introduce the following sets:

$$\begin{aligned} C &= \{1, \ldots, N_C\} \\ I &= \{1, 2\} \\ S[i] &= \{1, \ldots, N_S[p]\} \quad , i \in I \\ S_m[i] &= \{2, \ldots, N_S[i] - 1\} \quad , i \in I \end{aligned}$$

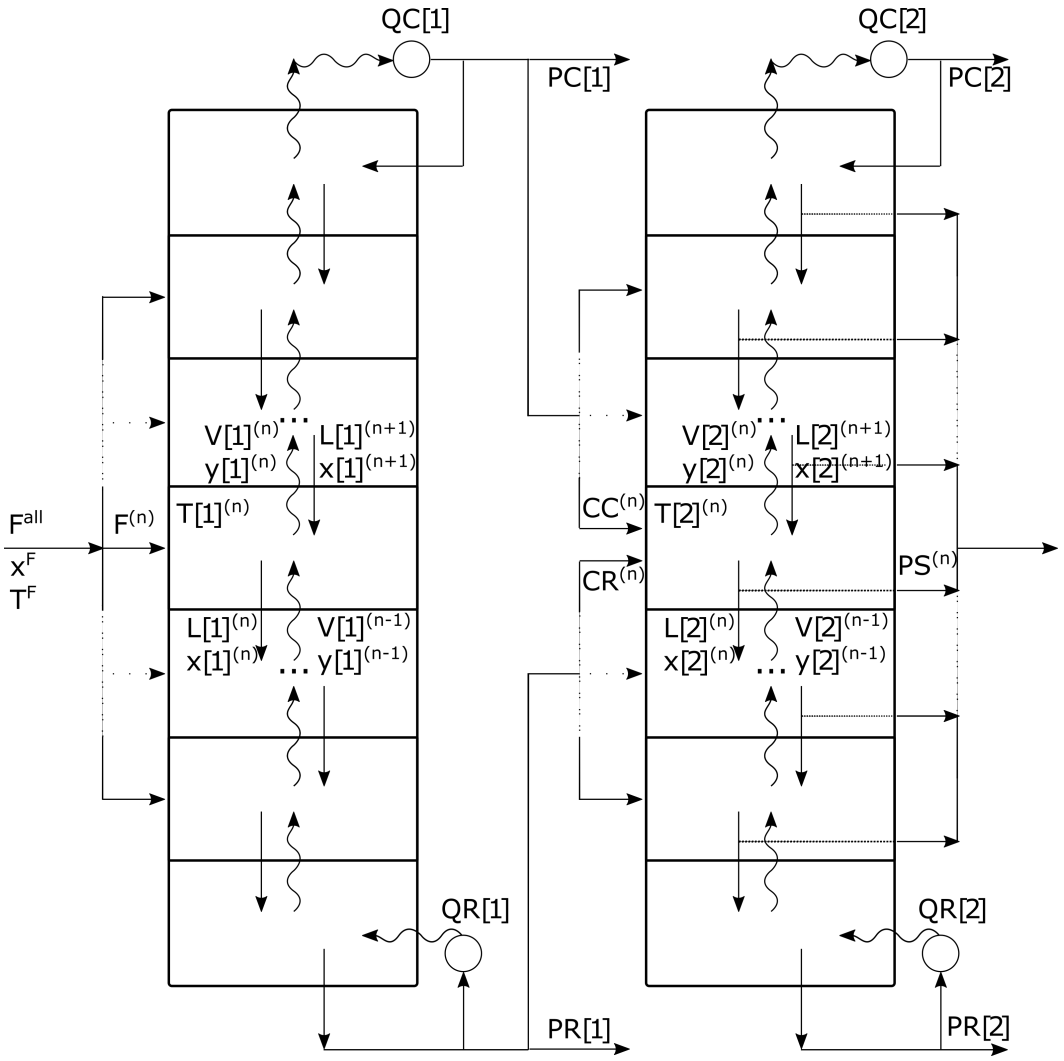


Figure 6.2: Superstructure for a three component feed mixture.

6.1.1 Constraints for distillation

6.1.1.1 Rigorous model

We generalize the equations arising from the rigorous model of a distillation column, see Section 2.2 to the superstructure. The variables introduced in Figure 6.2 have the following bounds:

$F^{(n)}$	$\geq 0,$	$n \in S_m[1]$
$V[i]^{(n)}, L[i]^{(n)}$	$\geq 0,$	$i \in I,\ n \in S[i]$
$y[i]_c^{(n)}, x[i]_c^{(n)}$	$\geq 0,\ \leq 1,$	$i \in I,\ n \in S[i],\ c \in C$
$CC^{(n)}, CR^{(n)}, PS^{(n)}$	$\geq 0,\ \leq F^{all},$	$n \in S_m[2]$
$PC[i], PR[i]$	$\geq 0, \leq F^{all},$	$i \in I$
$T[i]^{(n)}$	$\geq T_l, \leq T_u,$	$i \in I,\ n \in S[i]$
T^F	$\geq T_l, \leq T_u$	
$QC[i], QR[i]$	$\geq 0,$	$i \in I$

where we have chosen $T_l = 275$, $T_u = 500$.

The mass balances are similar to the single column case (2.16), but we need to incorporate the additional interconnecting streams. This gives for the first column:

$$
\begin{aligned}
0 =& V[1]^{(N_S[1]-1)}\ y[1]_c^{(N_S[1]-1)} - \left[L[1]^{(N_S[1])}\ x[1]_c^{(N_S[1])} + (PC[1] + \sum\nolimits_{n\in S_m[2]} CC^{(n)})\ y[1]_c^{(N_S[1])}\right] \\
0 =& V[1]^{(n-1)}\ y[1]_c^{(n-1)} + L[1]^{(n+1)}\ x[1]_c^{(n+1)} + F^{(n)}\ x_c^F - \left[V[1]^{(n)}\ y[1]_c^{(n)} + L[1]^{(n)}\ x[1]_c^{(n)}\right] \\
0 =& L[1]^{(2)}\ x[1]_c^{(2)} - \left[V[1]^{(1)}\ y[1]_c^{(1)} + (PR[1] + \sum\nolimits_{n\in S_m[2]} CR^{(n)})\ x[1]_c^{(1)}\right]
\end{aligned}
$$

with $c \in C, n \in S_m[1]$, where we have additional outgoing streams at the top and bottom stages; and for the second column:

$$
\begin{aligned}
0 =& V[2]^{(N_S[2]-1)}\ y[2]_c^{(N_S[2]-1)} - \left[L[2]^{(N_S[2])}\ x[2]_c^{(N_S[2])} + PC[2]\ y[2]_c^{(N_S[2])}\right] \\
0 =& V[2]^{(n-1)}\ y[2]_c^{(n-1)} + L[2]^{(n+1)}\ x[2]_c^{(n+1)} + CC^{(n)}\ y[1]_c^{(N_S[1])} + CR^{(n)}\ x[1]_c^{(1)} - \dots \\
& \dots \left[V[2]^{(n)}\ y[2]_c^{(n)} + L[2]^{(n)}\ x[2]_c^{(n)} + PS^{(n)}\ x[2]_c^{(n)}\right] \\
0 =& L[2]^{(2)}\ x[2]_c^{(2)} - \left[V[2]^{(1)}\ y[2]_c^{(1)} + PR[2]\ x[2]_c^{(1)}\right]
\end{aligned}
$$

with $c \in C,\ n \in S_m[2]$, where we have additional incoming streams at the intermediate stages.

The equilibrium equations from the single column case (2.17) are simply duplicated:

$$
\begin{aligned}
0 &= P\ y[i]_c^{(n)} - P_c^0(T[i]^{(n)})\ x[i]_c^{(n)}\ \gamma_c^(T[i]^{(n)}, x[i]^{(n)}) \\
0 &= P - \sum_{c\in C} x_c^F\ P_c^0(T^F)
\end{aligned}
$$

with $i \in I,\ n \in S[i], c \in C$.

The same holds for the summation equations (2.18):

$$
1 = \sum_{c\in C} y[i]_c^{(n)}, \quad 1 = \sum_{c\in C} x[i]_c^{(n)}
$$

with $i \in I,\ n \in S[i]$.

The heat balances (2.19) have to be adapted analogously to the mass balances. This yields for the first column:

$$
\begin{aligned}
0 =& \sum_{c\in C} \Big(V[1]^{(N_S[1]-1)}\ y[1]_c^{(N_S[1]-1)}\ H_c(T[1]^{(N_S[1])}) - \Big[L[1]^{(N_S[1])}\ x[1]_c^{(N_S[1])}\ h_c(T[1]^{(N_S[1])}) + \dots \\
& \dots \left(PC[1] + \sum\nolimits_{n\in S_m[2]} CC^{(n)}) y[1]_c^{(N_S[1])}\ h_c(T[1]^{(N_S[1])})\right]\Big) - QC[1] \\
0 =& \sum_{c\in C} \Big(V[1]^{(n-1)}\ y[1]_c^{(n-1)}\ H_c(T[1]^{(n-1)}) + L[1]^{(n+1)}\ x[1]_c^{(n+1)}\ h_c(T[1]^{(n+1)}) + F^{(n)}\ x_c^F\ h_c(T^F) - \dots \\
& \dots \left[V[1]^{(n)}\ y[1]_c^{(n)}\ H_c([1]T^{(n)}) + L[1]^{(n)}\ x[1]_c^{(n)}\ h_c(T[1]^{(n)})\right]\Big) \\
0 =& \sum_{c\in C} \Big(L[1]^{(2)}\ x[1]_c^{(2)}\ h_c(T[1]^{(2)}) - \Big[V[1]^{(1)}\ y[1]_c^{(1)}\ H_c(T[1]^{(1)}) + \dots \\
& \dots\ (PR[1] + \sum\nolimits_{n\in S_m[2]} CR^{(n)})\ x[1]_c^{(1)}\ h_c(T[1]^{(1)})\right]\Big) + QR[1]
\end{aligned}
$$

with $n \in S_m[1]$ and for the second column:

$$
\begin{aligned}
0 =& \sum_{c \in C} \Big(V[2]^{(N_S[2]-1)}\ y[2]_c^{(N_S[2]-1)}\ H_c(T[2]^{(N_S[2]-1)}) - \Big[L[2]^{(N_S[2])}\ x[2]_c^{(N_S[2])}\ h_c(T[2]^{(N_S[2])}) + \dots \\
& \dots\ PC[2]\ y[2]_c^{(N_S[2])}\ h_c(T[2]^{(N_S[2])}) \Big] \Big) - QC[2] \\
0 =& \sum_{c \in C} \Big(V[2]^{(n-1)}\ y[2]_c^{(n-1)}\ H_c(T[2]^{(n-1)}) + L[2]^{(n+1)}\ x[2]_c^{(n+1)}\ h_c(T[2]^{(n+1)}) + \dots \\
& \dots CC^{(n)}\ y[1]_c^{(N_S[1])}\ h_c(T[1]^{(N_S[1])}) + CR^{(n)}\ x[1]_c^{(1)}\ h_c(T[1]^{(1)})) - \dots \\
& \dots \Big[V[2]^{(n)} y[2]_c^{(n)}\ H_c(T[2]^{(n)}) + L[2]^{(n)}\ x[2]_c^{(n)}\ h_c(T[2]^{(n)}) + PS^{(n)}\ x[2]_c^{(n)}\ h_c(T[2]^{(n)}) \Big] \Big) \\
0 =& \sum_{c \in C} \Big(L[2]^{(2)}\ x[2]_c^{(2)}\ h_c(T[2]^{(2)}) - \Big[V[2]^{(1)}\ y[2]_c^{(1)}\ H_c(T[2]^{(1)}) + \dots \\
& \dots\ PR[2]\ x[2]_c^{(1)}\ h_c(T[2]^{(1)}) \Big] \Big) + QR[2]
\end{aligned}
$$

with $n \in S_m[2]$. Note, that all interconnecting and product streams are liquid, therefore are multiplied by the enthalpy of a liquid h_c.

6.1.1.2 Simplified model using constant molar overflow and constant relative volatilities

As mentioned before, the highly non-linear enthalpy, vapour pressure and activity coefficient equations can be hard to handle numerically. Therefore, we consider a simplified model, that still uses rigorous stage-wise MES balances, but utilizes the assumption of constant molar overflow to eliminate the heat balances, see Section 2.2.2.1. Further, it uses the assumption of constant relative volatilities to simplify the equilibrium equations to (2.7).

This approach eliminates the variables $T[i]^{(n)}$ (temperatures on each stage), $QC[i]/QR[i]$ (heat duties of condensers/reboilers) and the parameter P (pressure in the columns). Further, we eliminate the variables $V[i]^{(n)}, L[i]^{(n)},\ i \in I,\ n \in S[i]$ (mass flows between stages), since we can express all of them in terms of the new variables $V[i]^s, L[i]^s$ (mass flows in the stripping section):

The vapour flow between every stage equals $V[i]^s$, since it is assumed to be constant in each column section and the feed and all interconnecting streams are liquid. The liquid flow at some stage can be calculated by subtracting the mass flow that has entered in the stages below from $L[i]^s$, for example for the downwards liquid flow from stage n in the first column: $L[1]^s - \sum_{m=2}^{n-1} F^{(m)}$.

This gives the following mass balances for the first column:

$$
\begin{aligned}
0 =& V[1]^s\ y[1]_c^{(N_S[1]-1)} - \left[(PC[1] + \sum_{m \in S_m[2]} CC^{(m)})\ y[1]_c^{(N_S[1])} + (L[1]^s - F^{all})\ x[1]_c^{(N_S[1])} \right] \\
0 =& V[1]^s\ y[1]_c^{(n-1)} + (L[1]^s - \sum_{m=2}^{n} F^{(m)})\ x[1]_c^{(n+1)} + F^{(n)}\ x_c^F - \left[V[1]^s\ y[1]_c^{(n)} + (L[1]^s - \sum_{m=2}^{n-1} F^{(m)})\ x[1]_c^{(n)} \right] \\
0 =& L[1]^s\ x[1]_c^{(2)} - \left[V[1]^s\ y[1]_c^{(1)} + (PR[1] + \sum_{m \in S_m[2]} CR^{(m)})\ x[1]_c^{(1)} \right]
\end{aligned}
$$

with $n \in S_m[1],\ c \in C$;

and for the second column:

$$0 = V[2]^s\ y[2]_c^{(N_S[2]-1)} - \left[PC[2]\ y[2]_c^{(N_S[2])} + \left(L[2]^s - \sum_{m\in S_m[2]}\left[CC^{(m)} + CR^{(m)} - PS^{(m)}\right]\right)\ x[2]_c^{(N_S[2])}\right]$$

$$0 = V[2]^s\ y[2]_c^{(n-1)} + \left(L[2]^s - \sum_{m=2}^{n}\left[CC^{(m)} + CR^{(m)} - PS^{(m)}\right]\right)\ x[2]_c^{(n+1)} + CC^{(n)}\ y[1]_c^{(N_S[1])} + \dots$$

$$\dots CR^{(n)}\ x[1]_c^{(1)} - \left[V[2]^s\ y[2]_c^{(n)} + \left(L[2]^s - \sum_{m=2}^{n-1}\left[CC^{(m)} + CR^{(m)} - PS^{(m)}\right]\right)\ x[2]_c^{(n)} + PS^{(n)}\ x[2]_c^{(n)}\right]$$

$$0 = L[2]^s\ x[2]_c^{(2)} - \left[V[2]^s\ y[2]_c^{(1)} + PR[2]\ x[2]_c^{(1)}\right]$$

with $n \in S_m[2],\ c \in C$.

To ensure that we do not obtain negative liquid mass flow at some point in the superstructure, we need to introduce the following inequalities:

$$\begin{aligned} Ls[1] &\geq F^{all} \\ Ls[2] &\geq \sum_{m\in S_m[2]} CR^{(m)} \\ Ls[2] &\geq \sum_{m\in S_m[2]} (CC^{(m)} + CR^{(m)} - PS^{(m)}). \end{aligned}$$

The equilibrium equation simplify to the constant relative volatility model (2.7):

$$y[i]_c^{(n)}(\sum_{d\in C}\alpha_d x[i]_d^{(n)}) = \alpha_c x[i]_c^{(n)}$$

with $i \in I,\ n \in S[i],\ c \in C$.

The only equations that remain unchanged are the summation equations:

$$1 = \sum_{c\in C} y[i]_c^{(n)}, \quad 1 = \sum_{c\in C} x[i]_c^{(n)}$$

with $i \in I,\ n \in S[i]$.

6.1.2 Constraints for basic configurations

So far, we have obtained two sets of equations that model the distillation process within the superstructure, at different levels of simplifications. However, it would be feasible for all product and interconnecting streams to be non-zero.

Therefore, need to introduce additional constraints, that restrict the set of feasible solutions to the three column sequences pictured in Figure 6.1. They are characterized by the following properties:

- direct sequence: $CC^{(m)}, PS^{(m)} = 0,\ \forall m \in S_m[2]$ and $PR[1] = 0$
- indirect sequence: $CR^{(m)}, PS^{(m)} = 0,\ \forall m \in S_m[2]$ and $PC[1] = 0$
- prefractionator: $PR[1], PC[1] = 0$

In the following, we introduce three possibilities that each use the inherent structure of a different type of optimization problem to restrict the feasible set to the three cases above.

6.1.2.1 MINLP formulation

We introduce binary variables $y_{PC[1]}, y_{PR[1]}, y_{PS} \in \mathbb{B}$ that correspond to the three column sequences. For example, $y_{PC[1]} = 1$ corresponds to the direct sequence, which is the only case with $PC[1] > 0$. We need to ensure, that exactly one of the cases holds:

$$y_{PC[1]} + y_{PR[1]} + y_{PS} = 1. \tag{6.1}$$

Then, we enforce the corresponding properties by the following equations:

$$(1 - y_{PC[1]})PC[1] + y_{PC[1]} \sum_{m \in S_m[2]} CC^{(m)} = 0 \tag{6.2}$$

$$(1 - y_{PR[1]})PR[1] + y_{PR[1]} \sum_{m \in S_m[2]} CR^{(m)} = 0 \tag{6.3}$$

$$(1 - y_{PS}) \sum_{m \in S_m[2]} PS^{(m)} = 0 \tag{6.4}$$

If $y_{PC[1]} = 1$, then (6.2) implies $CC^{(m)} = 0,\ \forall m \in S_m[2]$. Since from (6.1) it follows, that $y_{PR[1]} = 0$ and $y_{PS} = 0$, we can conclude $PR[1] = 0$ from (6.3) and $PS^{(m)} = 0,\ \forall m \in S_m[2]$ from (6.4). The other two cases can be checked analogously.

Note, that since all terms that appear on the left hand side in (6.2)-(6.4) are non-negative, we can relax the equations to $\leq$-inequalities.

6.1.2.2 MPCC formulation

By using complementarity constraints, we can restrict the feasible set to the three column sequences without introducing additional variables:

$$0 \leq PC[1] \perp PR[1] \geq 0 \tag{6.5}$$

$$0 \leq \sum_{m \in S_m[2]} PS^{(m)} \perp PC[1] + PR[1] \geq 0 \tag{6.6}$$

$$0 \leq PC[1] \perp \sum_{m \in S_m[2]} CC^{(m)} \geq 0 \tag{6.7}$$

$$0 \leq PR[1] \perp \sum_{m \in S_m[2]} CR^{(m)} \geq 0 \tag{6.8}$$

The constraint (6.5) enforces, that at most one of the terms $PC[1]$ or $PR[1]$ is non-zero. Further, by (6.6) the term $\sum_{m \in S_m[2]} PS^{(m)}$ is non-zero, if and only if both $PC[1]$ and $PR[1]$ are zero. Therefore, the first two constraints enforce the exclusivity of the three cases and the corresponding conditions on the product streams: If $PC[1] > 0$, which means that the solution corresponds to a direct sequence, then $PR[1] = 0$ by (6.5) and $PS^{(m)} = 0,\ \forall m \in S_m[2]$ by (6.6). Additionally, the last two constraints enforce the conditions on the interconnecting streams corresponding to each case: If $PC[1] > 0$, then (6.7) implies $CC^{(m)} = 0,\ \forall m \in S_m[2]$. The other two cases can be checked analogously.

6.1.2.3 GDP formulation

The GDP formulation natively supports the disjunctive nature of the three cases. We introduce the Boolean variables $Y_{PC[1]},\ Y_{PR[1]},\ Y_{PS} \in \{True, False\}$, which can be interpreted as in the MINLP formulation, and write:

$$\begin{bmatrix} Y_{PC[1]} \\ \sum_{m \in S_m[2]} (CC^{(m)} + PS^{(m)}) + PR[1] \leq 0 \end{bmatrix} \bigvee \begin{bmatrix} Y_{PR[1]} \\ \sum_{m \in S_m[2]} (CR^{(m)} + PS^{(m)}) + PC[1] \leq 0 \end{bmatrix} \bigvee \begin{bmatrix} Y_{PS} \\ PR[1] + PC[1] \leq 0 \end{bmatrix}$$

If $Y_{PC[1]} = True$, then the corresponding inequality is enforced. Since all terms on the left hand-side are non-negative, this implies that every summand is zero, as intended.

6.1.3 Constraints for separation

So far, we have derived a number of formulations that model the distillation process within the superstructure, at different levels of simplification, and restrict the feasible set to the three column sequences corresponding to basic configurations, by different types of constraints. However, it would be feasible to obtain products of any compositions (that are physically feasible).
Therefore, we need to introduce additional constraints to ensure that the feed mixture is separated into (almost) pure products. Since, depending on the column sequence, the same component can be obtained at different product streams within the superstructure, we can not use explicit bounds on the concentrations of components in each product stream. Instead, we need to introduce a general measure for the quality of a separation.
The idea of our approach is the following: The bigger the differences between the concentrations of different components are, the "better" a product of a separation is. Let x be the composition of a product stream. We consider the sum of the squares of the above differences

$$\begin{aligned} \sum_{c \in C} \sum_{d \in C} (x_c - x_d)^2 &= \sum_{c \in C} \sum_{d \in C} (x_c^2 - 2x_c x_d + x_d^2) \\ &= \sum_{c \in C} N_C x_c^2 - 2 \left(\sum_{c \in C} x_c \right) \left(\sum_{d \in C} x_d \right) + \sum_{d \in C} N_C x_d^2 \\ &= 2N_C \sum_{c \in C} x_c^2 - 2 \end{aligned}$$

where we use the summation equation in the last step. Note, that it suffices to consider the term $\sum_{c \in C} x_c^2$, since the others are constant. We observe

$$\sum_{c \in C} x_c^2 \leq \left(\max_{c \in C} x_c \right) \left(\sum_{c \in C} x_c \right) = \max_{c \in C} x_c \leq 1.$$

This shows, that the term $\sum_{c \in C} x_c^2$ is maximal, if $x_d = 1,\ x_c = 0,\ \forall c \in C \backslash \{d\}$ for some $d \in C$, which corresponds to a pure product. Now, if we bound the term $\sum_{c \in C} x_c^2$ from below by a value close to 1, we enforce that we obtain an (almost) pure product.

To extend this idea to our superstructure, we can add up analogous terms for the compositions of all possible product streams and bound the sum from below. However, a solution that produces a large mass flow with composition close to the feed composition at $PR[2]$ and very pure products at $PC[1]$ and $PR[2]$, but with very small corresponding mass flows, would be feasible for large lower bounds. Obviously, we want to develop a constraint that excludes this kind of solution.
Therefore, we attach the mass flows of the product streams as weights. Let K be the index set of all product streams. Further, let P^k be the mass flow and x^k the composition of product stream $k \in K$. Then we consider:

$$Sep := \sum_{k \in K} P^k \sum_{c \in C} (x_c^k)^2 \leq \sum_{k \in K} P^k = F^{all}$$

Sep is maximal, if for each $d \in C$ we have $x_d^k = 1,\ x_c^k = 0,\ \forall c \in C \backslash \{d\}$ and $P^k = F^{all} x_d^F$ for some $k \in K$. For the remaining product streams we must have $P^k = 0$. This means, that for each component the mass flow contained in the feed is completely recovered as a pure product. Now, if we bound Sep from below by a value close to F^{all}, we enforce that we obtain (almost) pure products with mass flows close to the corresponding component mass flows in the feed.
Finally, we obtain the following constraint for our superstructure

$$\begin{aligned} \beta F^{all} \leq & PC[1] \sum_{c \in C} (y[1]_c^{(N_S[1])})^2 + PR[1] \sum_{c \in C} (x[1]_c^{(1)})^2 + \dots \\ & \dots PC[2] \sum_{c \in C} (y[2]_c^{(N_S[2])})^2 + PR[2] \sum_{c \in C} (x[2]_c^{(1)})^2 + \sum_{m \in S_m[2]} PS^{(m)} \sum_{c \in C} (x[2]_c^{(m)})^2 \end{aligned} \tag{6.9}$$

where the parameter β can be chosen on the interval $[0, 1]$.

6.2 Matrix-based approach

In this section, we derive in detail the problem formulation for finding the optimal distillation column sequence using a matrix-based approach and the Underwood model of a distillation column.
Firstly, we develop a method for enumerating all basic configurations based on the matrix approach, introduced in Section 4.2.2. Secondly, we modify the matrix approach to obtain a hybrid approach that yields constraints that can be used to restrict the feasible set to the basic configurations. Further, we generalize the Underwood model, established in Section 2.2.2.4, to the above formulations to obtain constraints that model the distillation.
The following derivations are valid for any value of N_C. To compare the formulation with the one resulting from the superstructure approach, we fix $N_C = 3$.

6.2.1 Constraints for basic configurations

We introduce the upper triangular matrix $U := \{(i,j) \in \{1,\dots,N_C\} \times \{1,\dots,N_C\} : i \leq j\}$ and the corresponding binary variables $X(i,j) \in \mathbb{B},\ (i,j) \in U$.

6.2.1.1 Enumeration with matrix approach

To enumerate all basic configurations that can separate feed mixtures of N_C components, we iterate through all settings of the free binary variables $X(i,j),\ j = 2,\dots,N_C-1,\ i = 1,\dots j$ and check, if they correspond to a basic configuration by using the constraints developed in Section 4.2.2.
We define

$$S^{(p)} := \{(i,j) \in U : \left\lfloor \frac{p}{2^{(i+\sum_{k=1}^{j-1} k-2)}} \right\rfloor \mod 2 = 1\},\ p = 0,\dots,2^{(N_C-2)(N_C+1)/2} - 1.$$

The sets $S^{(p)}$ are the elements of the power set of $U \backslash \{(1,1),(1,N_C),\dots,(N_C,N_C)\}$, which are the entries of the free binary variables.
Therefore, for each index $p = 0,\dots,2^{(N_C-2)(N_C+1)/2} - 1$, we set

$$X^{(p)}(i,j) := \begin{cases} 1, & (i,j) \in S^{(p)} \cup \{(1,1),(1,N_C),\dots,(N_C,N_C)\} \\ 0, & \text{else} \end{cases}.$$

and check (M1):

$$\forall j = 2,\dots,N_C,\ i = 1,\dots,j : \sum_{m=1}^{j-i} X^{(p)}(i,j-m) + \sum_{m=1}^{i-1} X^{(p)}(i-m,j-m) \geq X^{(p)}(i,j)$$

and (M2):

$$\forall j = 1,\dots,N_C-2,\ i = 1,\dots,j,\ X^{(p)}(i,j) = 1 :$$
$$\min\{m \in M : X^{(p)}(i,j+m) = 1\} + \min\{m \in M : X^{(p)}(i+m,j+m) = 1\} + j \leq N_C + 1$$

with $M := \{1,\dots,N_C - j\}$.

To speed up the process, it is useful to also introduce the preliminary check:

$$\sum_{j=2}^{N_C-1} \sum_{i=1}^{j} X^{(p)}(i,j) \geq N_C - 2.$$

Any feasible distillation configuration includes at least $N_C - 1$ splits and therefore at least $N_C - 2$ submixtures (excluding the N_C pure components).
If one of these checks fails, we break the procedure and forget the candidate $X^{(p)}$. If all of them are successful, we save it. We obtain a list of all basic configurations, described by the variables X. Specifically, we can calculate their number by counting the entries of the list.

6.2.1.2 Simultaneous optimization with hybrid approach

We modify the matrix approach by introducing additional variables $Y(i,j,l),\ Z(i,j,l) \in \mathbb{B},\ (i,j,l) \in E$ with $E := \{(i,j,l) \in \{1,\dots,N_C\} \times \{1,\dots,N_C\} \times \{1,\dots,N_C\} : i \leq j < l\}$.
These variables model the arrows in the matrix approach, that point from a feed to its products and thereby specify the distillation configuration corresponding to X, see for example (4.2). There exists a (horizontal) arrow from (i,j) to (i,l), if and only if $Y(i,j,l) = 1$. Similarly, there exists a (diagonal) arrow from (i,j) to $(i+(l-j),l)$, if and only if $Z(i,j,l) = 1$.
Note, that this also gives an interpretation in terms of the graph model: Y and Z correspond to variables on the edges, while X corresponds to variables on the nodes.
We use these interpretations to derive a novel formulation, which we call *hybrid approach*:
If a submixture, except the pure components, ($j = 1,\dots,N_C-1,\ i = 1,\dots,j$) exists (such that $X(i,j) = 1$), there exists exactly one distillate product ($\exists!\ l \in \{j+1,\dots,N_C\} : Y(i,j,l) = 1$) and one bottom product ($\exists!\ l \in \{j+1,\dots,N_C\} : Z(i,j,l) = 1$). If a submixture does not exists (such that $X(i,j) = 0$), there exists no product ($\forall\ l \in \{j+1,\dots,N_C\} : Y(i,j,l) = 0,\ Z(i,j,l) = 0$):

$$\begin{aligned} \forall\ j = 1,\dots,N_C-1,\ i = 1,\dots,j: &\quad \sum_{l=j+1}^{N_C} Y(i,j,l) = X(i,j) \\ \forall\ j = 1,\dots,N_C-1,\ i = 1,\dots,j: &\quad \sum_{l=j+1}^{N_C} Z(i,j,l) = X(i,j) \end{aligned} \tag{H1}$$

Similarly, we derive the following two constraints:
If a submixture is produced, it must exist:

$$\begin{aligned} \forall\ j = 2,\dots,N_C,\ i = 1,\dots,j: &\quad \sum_{l=i}^{j-1} Y(i,l,j) \leq X(i,j) \\ \forall\ j = 2,\dots,N_C,\ i = 1,\dots,j: &\quad \sum_{l=j-i+1}^{j-1} Z(l-(j-i),l,j) \leq X(i,j) \end{aligned} \tag{H2}$$

If a submixture, except the feed mixture, exists, it must be produced:

$$\forall\ j = 2,\dots,N_C,\ i = 1,\dots,j: \quad \sum_{l=i}^{j-1} Y(i,l,j) + \sum_{l=j-i+1}^{j-1} Z(l-(j-i),l,j) \geq X(i,j) \tag{H3}$$

Note, that (H1)-(H3) combine to "Ensure mass balance" and "Ensure that pure components are made" and are equivalent with (M1).

The number of components of every submixture, that exists, must be less or equal the sum of the numbers of components of its products:

$$\forall\, j = 1, \dots, N_C - 2,\ i = 1, \dots, j: \quad \sum_{l=j+1}^{N_C} (Y(i,j,l) + Z(i,j,l))(N_C + 1 - l) \geq X(i,j)(N_C + 1 - j) \tag{H4}$$

This constraint expresses "Ensure that no components vanish" and is equivalent with (M2).

We obtain a formulation with the following number of free variables:

$$\begin{aligned}\sum_{j=2}^{N_C-1} j + 2\sum_{l=1}^{N_C}\sum_{j=1}^{l-1} j &= \frac{(N_C+1)(N_C-2)}{2} + \frac{N_C(N_C+1)(2N_C-1)}{6} - \frac{N_C(N_C+1)}{2} \\ &= (N_C+1)\frac{3(N_C-2) + N_C(2N_C+1) - 3N_C}{6} \\ &= \frac{(N_C+1)(N_C+2)(2N_C-3)}{6} = \mathcal{O}(N_C^3)\end{aligned}$$

and number of constraints:

$$\begin{aligned}2\sum_{j=1}^{N_C-1} j + 3\sum_{j=2}^{N_C} j + \sum_{j=1}^{N_C-2} j &= 2\frac{(N_C-1)N_C}{2} + 3\left(\frac{N_C(N_C+1)}{2} - 1\right) + \frac{(N_C-2)(N_C-1)}{2} \\ &= (N_C-1)\frac{2N_C + 3(N_C+2) + (N_C-2)}{2} = (N_C-1)(3N_C+2) = \mathcal{O}(N_C^2).\end{aligned}$$

We have obtained a system of linear equations and inequalities that restricts the feasible set to the basic configurations. Specifically, we have replaced the non-smooth inequalities (M2) in the matrix approach by the linear inequalities (H4). In Appendix C we rigorously proof the equivalence of the hybrid and matrix formulations. Additionally, in Appendix D we derive a generalization of the original graph approach by using the results from this section.

6.2.2 Constraints for distillation

In this section, we discuss the additional variables and constraints that arise from the Underwood model for a distillation column, established in Section 2.2.2.4. Note, that in this derivation we assume perfect separations. It can be extended to the case of impure products by adapting the Underwood equations for the key components.

Consider a basic configuration described by the variables $X(i,j),\ Y(i,j,l),\ Z(i,j,l) \in \mathbb{B},\ (i,j) \in U,\ (i,j,l) \in E$ as introduced in the hybrid approach.

Then, the light and heavy key components of the split, where the submixture corresponding to

$(i,j) \in U,\ j \leq N_C - 1$ acts as feed, are given by:

$$LK(i,j) = i + \sum_{k=1}^{N_C - j} Z(i,j,j+k)\ k - 1$$
$$HK(i,j) = i + NC - j - \sum_{k=1}^{N_C - j} Y(i,j,j+k)\ k + 1.$$

This can be derived as follows: If $Z(i,j,j+k) = 1$, then the bottom product of the split is the submixture corresponding to $(i+k, j+k)$, which is $i+k \cdots i+k+(N_C-(j+k))$. Therefore the lightest component that is not in the bottom product, i.e. the light key component, is $i+k-1$. Analogously, if $Y(i,j,j+k) = 1$, the top product is the mixture corresponding to $(i, j+k)$, which is $i \cdots i+(N_C-(j+k))$; this means, $i + N_C - j - k + 1$ is the heavy key component.

We introduce the following variables with bounds:

$$\begin{array}{lll} V(i,j) & \geq 0, & (i,j) \in U,\ j \leq N_C - 1 \\ D(i,j,l,c) & \geq 0, \leq F^{all}\ Y(i,j,l), & (i,j,l) \in E,\ c \in C \\ B(i,j,l,c) & \geq 0, \leq F^{all}\ Z(i,j,l), & (i,j,l) \in E,\ c \in C \end{array}$$

$V(i,j)$ is the vapour flow in the column section that performs the split, where the submixture corresponding to (i,j) acts as feed. $D(i,j,l,c),\ B(i,j,l,c)$ give the distillate and bottom product flows of component c from the aforementioned column section to the column sections that perform the splits, where the mixtures corresponding to (i,l) and $(i+(l-j),l)$, respectively, act as feed.

The following material balances hold:

$$0 = F^{all}\ x_c^F - \left(\sum_{l=2}^{N_C} D(1,1,l,c) + \sum_{l=2}^{NC} B(1,1,l,c) \right)$$
$$0 = \sum_{l=i}^{j-1} D(i,l,j,c) + \sum_{l=j-i+1}^{j-1} B(l-(j-i),l,j,c) - \left(\sum_{l=j+1}^{N_C} D(i,j,l,c) + \sum_{l=j+1}^{N_C} B(i,j,l,c) \right)$$

with $(i,j) \in U,\ 2 \leq j \leq N_C - 1,\ c \in C$.

Since we assume perfect separations, all components lighter or equal the light key do not appear in the bottom product. Similarly, all components heavier or equal the heavy key do not appear in the distillate product. Therefore, we fix:

$$\sum_{l=2}^{N_C} B(1,1,l,c) = 0, \qquad c = 1, \ldots, LK(1,1) \tag{6.10}$$
$$\sum_{l=j+1}^{N_C} B(i,j,l,c) = 0, \qquad (i,j) \in U,\ 2 \leq j \leq N_C - 1, \qquad c = 1, \ldots, LK(i,j)$$
$$\sum_{l=2}^{N_C} D(1,1,l,c) = 0, \qquad c = HK(1,1), \ldots, N_C$$
$$\sum_{l=j+1}^{N_C} D(i,j,l,c) = 0, \qquad (i,j) \in U,\ 2 \leq j \leq N_C - 1, \qquad c = HK(i,j), \ldots, N_C$$

We introduce the r-th Underwood root of the split, where the mixture corresponding to (i,j) acts as feed:

$$\alpha_{r+1} \leq \phi(i,j,r) \leq \alpha_r$$

with $(i,j) \in U,\ j \leq N_C - 1,\ r = i, \ldots, i + N_C - j - 1$.
Then the Underwood feed equation for the first split, compare (2.28), is given by:

$$\sum_{c \in C} \frac{\alpha_c F^{all} x_c^F}{\alpha_c - \phi(1,1,r)} = 0$$

with $r = 1, \ldots, N_C - 1$.
Similarly, the Underwood feed equations for the remaining splits are:

$$\sum_{c \in C} \frac{\alpha_c \ \left(\sum_{l=i}^{j-1} D(i,l,j,c) + \sum_{l=j-i+1}^{j-1} B(l-(j-i),l,j,c) \right)}{\alpha_c - \phi(i,j,r)} = 0$$

with $(i,j) \in U,\ 2 \leq j \leq N_C - 1$ such that $X(i,j) = 1$ (else the column section does not exist) and $r = i, \ldots, i + N_C - j - 1$.
The Underwood equation for the rectifying section, compare (2.26), becomes:

$$V(i,j) \geq \sum_{c \in C} \frac{\alpha_c \ \sum_{l=j+1}^{N_C} D(i,j,l,c)}{\alpha_c - \phi(i,j,r)}$$

with $(i,j) \in U,\ j \leq N_C - 1$ such that $X(i,j) = 1$ and $r = LK(i,j), \ldots, HK(i,j) - 1$.
Note, that the last constraint is an inequality, since by integrating multiple splits into one distillation column, not every column section must operate at the optimum (minimum reflux). Instead, we must introduce additional coupling constraints:

$$V(i,l_y) = V(l_z - (j-i), l_z) \tag{6.11}$$

$$D(i,l_y,j,c) \left(\sum_{d \in C} \alpha_d \ B(l_z - (j-i), l_z, j, d) \right) = \alpha_c \ B(l_z - (j-i), l_z, j, c) \ \sum_{d \in C} D(i,l_y,j,d) \tag{6.12}$$

with $(i,j) \in U,\ 2 \leq j \leq N_C - 1$ such that $X(i,j) = 1$ and $l_y = i, \ldots, j-1,\ l_z = j-i+1, \ldots, j-1$ such that $Y(i,l_y,j) = 1 \wedge Z(l_z - (j-i), l_z, j) = 1$.
This can be derived as follows: If two column sections produce the same product (which is described by the fact, that its corresponding matrix entry has both a horizontal and a diagonal incoming edge), then they must have the same vapour flow and their product streams must be in equilibrium.
We must also adapt our objective, in order to count the vapour flow of a distillation column containing multiple splits only once:

$$\min: \sum_{(i,j) \in U, j \leq N_C - 1} V(i,j) - \sum_{\substack{(i,j) \in U, j \leq N_C - 1, \\ l_y = i, \ldots, j-1, l_z = j-i+1, \ldots, j-1: \\ X(i,j) = 1 \wedge Y(i,l_y,j) = 1 \wedge Z(l_z - (j-i), l_z, j) = 1}} V(l_z - (j-i), l_z)$$

The derived formulation can be applied directly to the enumeration approach, where in each iteration the values of X are fixed. Since two entries are connected by an arrow, if and only if they both have value 1 and none of the entries on their connecting line have value 1, we calculate the corresponding values of Y, Z

$$Y(i,j,l) = X(i,j)\ X(i,l)\ \max(0, 1 - \sum_{m=j+1}^{l-1} X(i,m))$$

$$Z(i,j,l) = X(i,j)\ X(l-(j-i),l)\ \max(0, 1 - \sum_{m=j+1}^{l-1} X(m-(j-i),m))$$

with $(i,j,l) \in E$. The minimum of the resulting NLP, given by the constraints introduced above, gives the minimum vapour flow of the basic configuration in each iteration.
In case of the simultaneous optimization approach, X, Y, Z and therefore also LK, HK are variables. In order to avoid logical constraints, we must reformulate the above constraints slightly, e.g for the first separation-constrained component balance (6.10):

$$\max(LK(1,1) - c + 1, 0) \sum_{l=2}^{N_C} B(1,1,l,c) = 0, c \in C$$

or for the first coupling constraint (6.11):

$$X(i,j)Y(i,l_y,j)Z(l_z-(j-i),l_z,j)(V(i,l_y) - V(l_z-(j-i),l_z)) = 0$$

with $(i,j) \in U, 2 \leq j \leq N_C - 1$ and $l_y = i, \ldots, j-1, l_z = j-i+1, \ldots, j-1$. The global minimum of the resulting MINLP gives the basic configuration with the lowest minimum vapour flow.

Chapter 7

Technical framework

In this chapter, we discuss the technical framework we use to numerically solve the optimization problems formulated in the last chapter. Many algorithms have been developed and analysed to solve the types of optimization problems we have established reliably and quickly.
The implementation of these, however, would go beyond the scope of this thesis. Therefore, we have chosen to utilize the *NEOS* (Network-Enabled Optimization System) *Server*, a free internet-based service, that provides access to over 60 state-of-the-art solvers for a variety of types of optimization problems. We explain different ways to use the services provided by the server and discuss the corresponding advantages and disadvantages.

7.1 NEOS Server

The NEOS Server workflow can be organized into the following five steps [51]:

1. Based on the type of optimization problem, select an appropriate solver.
2. For the selected solver, choose an appropriate input format and prepare the model, data and additional files as specified.
3. Select a method of submission for your job: web interface, email, XML-RPC or Kestrel.
4. Submit your job to the NEOS Server. When the NEOS Server receives the submission, it assigns a job number and a password and queues the job for scheduling on one of the machines.
5. When the job is finished, retrieve the results via the job number and password on the web interface or directly, when using other methods of submission.

The available solvers are partly commercial products, which are provided free-of-charge by the developing companies, and partly NEOS Server implementations of academic codes or open source codes. This means the NEOS Server is a unique way to test, compare and use a variety of state-of-the-art solvers for free and without the need for any software.
The NEOS Server is hosted by the Wisconsin Institutes of Discovery at the University of Wisconsin in Madison and the jobs run on distributed high-performance machines provided by the Center for High Throughput Computing (CHTC) at the University of Wisconsin, the Arizona State University, the University of Klagenfurt in Austria, and the University of Minho in Portugal.

Details on the design and implementation of the NEOS Server can be found in [52], while an overview is given in [53]. Additional information about the development, usage and future challenges is collected in [54].

7.1.1 Solvers

The types of optimization problems discussed in the previous chapter are: non-linear program (NLP), mixed-integer non-linear program (MINLP), mathematical program with complementarity constraints (MPCC) and generalized disjunctive program (GDP). The solvers available on the NEOS Server for solving each of these types are given in Table 7.1. We have also included solvers for solving optimization problems globally, which is generally of interest.
Most of these solvers demand input in AMPL or GAMS format, which are the two most common algebraic modelling languages for describing and solving large-scale mathematical optimization problems. We have indicated for each solver which of the two formats it supports in Table 7.1.

7.2 Modelling languages

Algebraic modelling languages (AML) are high-level computer programming languages for describing and solving high complexity problems for large scale mathematical computation. Their distinguishing features are: One, the use of relational algebra to support languages elements such as sets and indices, which enables a syntax that is very similar to the notation used in optimization problems and therefore leads to very readable code. And two, the support of automatic differentiation to obtain partial derivatives on very large structures. [56]
The most established two AML, that are also supported by the NEOS Server, are AMPL and GAMS. We introduce their syntax for formulating an optimization problem in the following.
In order to submit the required input to the NEOS Server via the web interface, it is sufficient to write the problem formulation in the appropriate (language-specific) syntax in a simple text editor. However, for both languages there exist corresponding IDEs which enable text highlighting, to simplify creating the input, and include a compiler, to test the created files on correct syntax and completeness. Further, they enable a direct connection to the NEOS Server within the IDE via the interface Kestrel.

7.2.1 AMPL

A Mathematical Programming Language (AMPL) was first developed by Robert Fourer, David Gay, and Brian Kernighan at Bell Laboratories in 1985, its last stable release was in 2013.
A free demo version of the IDE including the language compiler and the interface to the NEOS Server Kestrel can be downloaded at `https://ampl.com/try-ampl/download-a-free-demo/`. It supports the creation of models with up to 500 variables (300 for programs with non-linear constraints). Further, it includes demo versions of the following solvers: `CONOPT`, `LOQO`, `MINOS`, and `SNOPT`; and `Knitro` and `BARON` (for up to 10 variables). This means, these solvers can be accessed directly within the IDE, without using the NEOS Server.
An AMPL description of an optimization problem can include up to 3 different file types:

The key element is the *model file* (.mod), which includes the declaration (and possible assignment) of variables, parameters, constraints and objectives. Every formulation of an optimization problem in AMPL must include exactly one model file.
Additionally, we can include arbitrarily many *data files* (.dat). Here, we can assign values to the sets and parameters declared in the model file, which we want to easily change between optimization runs. For example, the mixture-specific parameters (relative volatilities or parameters for vapour pressures/activity coefficients/enthalpies) can be stored in separate data files, so that we only have to change the reference to the data file and not the model file when considering different mixtures.
The optimization is tied together by the *commands file* (.run). This must include commands to specify the model and (optional) data file(s) describing the optimization problem. It can include additional declarations of variables, constraints or objectives and additional specifications of data, that are specific to runs of the optimization problem. Lastly, it must include the command "solve;" which starts the connection to solver. Displaying the obtained results or writing them into files can also be introduced in this file.
In the following we consider a simple general optimization problem, to introduce main aspects of the syntax that are integral for our optimization problems at hand.

$$\begin{aligned}
&\min \sum_{s\in S}\sum_{t\in T} z(s,t) \\
&\sum_{t\in T} y(t) = 1 \\
&\sum_{t\in T} x(s)\exp(q(s)z(s,t) + r(s,t)y(t)) = p, \quad && s\in S \\
&x(s) \in \mathbb{R}, && s\in S \\
&y(t) \in [0,1], && t\in T \\
&z(s,t) \in \mathbb{Z}, && s\in S,\ t\in T
\end{aligned} \tag{7.1}$$

with parameters $p,\ q(s),\ s\in S$ and $r(s,t),\ s\in S,\ t\in T$.

Model file Declaration of sets that act as index sets for parameters, variables and constraints:

```
set S := 1..5;
set T;
```

Note, that we can choose between directly assigning values to the sets in the model file, as in the first line, or only declaring them, as in the second line. In the latter case, we need to assign values in subsequent data statements. Operations on sets such as union, intersection etc. are supported.
Declaration of parameters, that are fixed in the optimization, but are possibly changed between different runs of optimization:

```
param p := 5;
param q {S};
param r {S,T};
```

Note, that we use the previously declared index sets to simplify the code.

Declaration of variables:

```
var x {S};
var y {T} >= 0, <= 1;
var z {S,T} integer;
```

Note, that we can declare bounds on variables or specify their types directly in their declaration.
Declaration of dependent variables:

```
var u {s in S, t in T} = exp(q[s]*z[s,t]+r[s,t]*y[t]);
```

This construct can be used to define terms that occur in multiple places and therefore create more readable code.
Declaration of constraints:

```
subject to Con1: sum{t in T} y[t] = 1;
subject to Con2 {s in S}: sum{t in T}(x[s]*u[s,t]) = p
```

Declaration of objective(s):

```
minimize Obj1: sum{s in S} sum_{t in T} z[s,t];
```

We save all the above code in a file called "example.mod".

Data file In the data file, we now assign values to the sets and parameters that were only declared in the model file.
Assignment of sets:

```
set T := 1 , 2 , 3;
```

Assignment of values to indexed parameters:

```
param q := 1 0.2 2 0.4 3 -0.3 4 0.3 5 0.1;
param r
:  1     2     3 :=
1  1.3   -2.7 1.8
2  -7.3  -5.6 -6.2
3  -20.6 -7.1 -9.2
4  0.02  0.1   0.4
5  1     2     2;
```

Note, that we can use a matrix notation to simplify assigning multidimensional data.
We save all the above code in a file called "example1.dat".

Command file In the command file, we now bring together the formulation of the optimization problem and submit it to a solver with the following statements:

```
model 'example.dat';
data 'example1.dat';
solve;
```

This is the minimal working example of a command file, which we save in a file called "example.run". To execute the optimization in the IDE, we simply write the command "include example.run" in the console. Note, that for each solve command, a link to the appropriate solver will be established and after it has finished an execution statement with information such as optimality criteria, number iterations etc. (specific to each solver) will be displayed in the console.
We can also change settings, modify our model and add output commands:

```
option solver 'Knitro';
model 'example.dat';
data 'example1.dat';
param p2;
redeclare subject to Con1: sum{t in T} y[t] =p2;
data;
param p2 := 2;
solve;
display _varname, _var > 'results.txt';
display _conname, _con.slack;
```

The first line sets the solver, that should be used for this optimization problem.
Then, in the same syntax as in the model file, we declare an additional parameter. By preceding the declaration with the keyword "redeclare", we can change an already declared constraint.
To include an additional data statement, we precede it with the command "data;". Then we can use the same syntax as in the data file.
After the execution of the solver, the values of the variables at the returned solution will be written into the file "results.txt" and the values of the slack of the constraints will be displayed in the console. Note, that it is possible to combine multiple "solve;" commands into one .run file and modify the model and/or data in between.

Submitting a job to the NEOS Server via Kestrel To avoid the limitations on the solvers of the free demo version of AMPL, for example only up to 10 variables for the `Knitro solver`, we submit the optimization problem to the NEOS Server. This can be done directly in the IDE via the interface Kestrel, which is invoked by including the following two commands in the command file:

```
option solver 'kestrel';
option kestrel_options 'solver=Knitro';
```

Now, for each solve command in the command file, a connection to the NEOS Server will be established and a jobnumber and password assigned, which are displayed in the console. When the NEOS Server obtains the results from the solver, they are returned to the IDE and the same execution statement, as when running the solver directly in the IDE, is displayed.
Overall, this method does not create any further restrictions on the workflow and therefore is very straightforward and efficient.
The only possible error is evoked, when there is a time-out of the connection to the NEOS Server. In that case, there is an error message displayed in the console, and the whole execution of the command file is aborted. It is possible to access the results of the current job on the NEOS Server by writing in the console

```
option kestrel_options job=* password=*
```

with the corresponding jobname and password.
However, this method of submission, using the demo version of the AMPL IDE, is limited to problems that do not exceed the size restrictions mentioned in the beginning.

Submitting a job to the NEOS Server via the web interface For larger models, the input files (model, data, commands) have to submitted directly to the NEOS Server. This can be done by e-mail or a web interface, we have chosen the latter in this work.
For each solver available on the NEOS Server, there exists a separate submission form, that can be accessed via the overview page `https://neos-server.org/neos/solvers/index.html`.
There are some restrictions that have to be considered:

- It is not possible to upload more than one data file, if a problem formulation contains multiple ones, they have to be stringed together.
- The "model *.mod" and "data *.dat" statements from the command file must be deleted, since the submission form names the uploaded files automatically.
- The interface can not produce output files. The command file must include "display" commands and the results must be saved from the results page by "copy and paste".

Handling MPCCs and GDPs Complementarity constraints are supported natively in the AMPL framework, for example $0 \leq x_1 \perp x_2 \geq 0$ can be introduced as constraint:

```
subject to Con: 0 <= x1 complements x2 >= 0$
```

where x, y have to be declared previously as variables.
Disjunctive programs, however, can not be expressed directly in the AMPL language. We have to reformulate them, as described in Section 3.4.1.

7.2.2 GAMS

General Algebraic Modeling System (GAMS) was developed by a research group at the World Bank in 1976, its last stable release was in 2018.
A free demo version of the IDE including the language compiler and the interface to the NEOS Server Kestrel can be downloaded at `https://www.gams.com/download/`. It supports the creation of models with up to 300 variables (up to 50 discrete variables) or constraints with up to 2000 non-zero elements (of which up to 1000 non-linear). Note, that for our optimization problem, the last restriction is much stricter than for the limitations of the AMPL IDE. It also includes demo versions of a number of solvers, for global solvers an additional restriction (up to 10 variables) applies.
A GAMS description of an optimization problem usually consists of one file, the main file (.gms). This includes both the model statements and commands. Optionally, data can be stored separately in data files (.gdx) and loaded.
We give the GAMS description of the optimization problem (7.1), analogous to the AMPL example:

Data file Firstly, we need to create the data file "example1.gdx" which stores the values of the set T and the parameters r, s. This can be done automatically by executing the following supplementary file "example1data.gms" in the IDE:

```
set T /1*3/;
parameter q /1 0.2 2 0.4 3 -0.3 4 0.3 5 0.1/;
table r(S,T)
     '1'   '2'   '3'
'1'  1.3   -2.7  1.8
'2'  -7.3  -5.6  -6.2
'3'  -20.6 -7.1  -9.2
'4'  0.02  0.1   0.4
'5'  1     2     2;

$gdxout example1
$unload S,q,r
$gdxout
```

Main file Now the main file "example.gms" includes the description of the optimization problem and the commands for solving it:

```
$gdxin example1.gdx
set S;
set T /1*5/;
$load S;
parameter p /5/;
parameter q(S);
parameter r(S,T);
$load q, r
$gdxin

variable x(S);
variable y(T);
y.lo = 0;
y.up = 1;
integer variable z(S,T);

equation Con1;
Con1 .. sum(T,y(T)) =e= 1;
equation Con2(S);
Con2(S) .. sum(T,x(S)*exp(q(S)*z(S,T)+r(S,T)*y(T)) =e= p;

variable Obj;
equation DefObj;
DefineObj .. Obj =e= sum(S,sum(T, z(S,T)));

Model example / Con1, Con2, DefObj /;

solve example using MINLP minimizing Obj;
```

Firstly, the data file is specified. After the declaration of the sets, we need to load the data for S, before declaring the parameters (dependent on S) and loading their data. The subsequent command stops the reading of the data file.
Then, the variables and their lower and upper bounds or variable type are declared. Note, that GAMS does not support the direct declaration of dependent variables.
Further, the constraints are declared and defined. Note, that inequality constraints may be declared by using "=l=" (less than or equal) or "=g=" (greater than or equal).
Lastly, the objective is declared as a new variable and defined via an equality constraint.
The creation of the model is concluded by naming it an specifying the involved constraints.
The solve command, specifying the type of optimization problem handled and the objective to be optimized, initiates the connection to the solver.
When executing "example.gms" in the IDE by pressing "F9", a new window with the execution statement of the solver is opened. Additionally, a new file (example.lst) listing the described model and the returned solution is created and displayed.

Submitting ta job to the NEOS Server via Kestrel To avoid the size limitation of the solvers included in the demo version of the IDE, we can submit a job to the NEOS Server via Kestrel, by adding the statements

```
option MINLP = kestrel;
columnsMINLP.optfile = 1;
```

before the solve command. We also need to create an additional .opt file in the same directory, which specifies the desired solver:

```
kestrel_solver 'Knitro'
neos_server neos-server.org:3333
```

Submitting a job to the NEOS Server via the web interface For problems exceeding the size limitations of the demo version, we use the web interface of the NEOS Server. Again, the submission forms for each solver can be accessed at `https://neos-server.org/neos/solvers/index.html`. There, we upload the main file and (optional) data file. Note, for any call of the data file in the main file, it must be renamed to "in.gdx".
It is possible to obtain the results returned from the solver as a downloadable data file (.gdx).

Handling MPCCs and GDPs The GAMS language supports MPCCs natively, by specifying complementarity constraints as equation-variable pairs, separated by a dot in the model declaration. For example, the toy problem:

$$\begin{aligned} \text{minimize} \quad & x_1 + 2x_2 \\ \text{subject to} \quad & -x_1 - x_2 \leq -1 \\ & 0 \leq x_1 \perp x_2 \geq 0 \end{aligned}$$

with optimal solution $\bar{x} = (1, 0)^T$, can be expressed as:

```
variable x1;
positive variable x2;
equation Con1;
Con1 .. -x1-x2 =l= -1;
equation Con2;
Con2 .. x1 =g= 0;
variable Obj;
equation DefObj;
DefObj .. Obj =e= x1 + 2*x2;
model example /DefObj, Con1, Con2.x2/;
solve example using MPEC minimizing Obj;
```

Further, the automatic reformulation of GDPs is supported via the Extended Mathematical Programming (EMP) extension. To use this, we must introduce an additional EMP info file "emp.info". Note, that we can create and write this file by adding additional commands to the main file, which is often most convenient. For example, the toy problem:

$$\begin{aligned} \text{minimize} \quad & x_1 + 2x_2 \\ \text{subject to} \quad & x_1, x_2 \geq 0 \\ & \begin{bmatrix} Y_1 \\ x_2 - x_1 + 2 \leq 0 \end{bmatrix} \bigwedge \begin{bmatrix} \neg Y_1 \\ 2 - x_2 \leq 0 \end{bmatrix} \\ & \begin{bmatrix} Y_2 \\ x_1 - x_2 \leq 0 \end{bmatrix} \bigwedge \begin{bmatrix} \neg Y_1 \\ x_1 - 1 \leq 0 \end{bmatrix} \\ & (Y_1 \wedge \neg Y_2) \vee (\neg Y_1 \wedge Y_3) \end{aligned}$$

with optimal solution $\bar{x} = (0, 2)^T$, can be expressed as:

```
positive variables x1,x2;
binary variables y1,y2;
equations Con1,Con2,Con3,Con4;
Con1.. x2 - x1 + 2 =l= 0;
Con2.. 2 - x2 =l= 0;
Con3.. x1 - x2 =l= 0;
Con4.. x1 - 1 =l= 0;
logic equation lCon1;
lCon1 .. (y1 and not y2) or (not y1 and y2);
variable Obj;
equation DefObj;
defobj.. Obj =e= x1 + 2*x2;
model example / all /;

File emp / '%emp.info%' /;
putclose emp
'disjunction y1 Con1 else Con2'
'disjunction y2 Con3 else Con4' ;

solve example using EMP minimizing Obj;
```

Type of optimization problem	Name of Solver	AMPL	GAMS
NLP	ANTIGONE		x
	CONOPT	x	x
	filter	x	
	Ipopt	x	x
	Knitro	x	x
	LANCELOT	x	
	LOQO	x	
	MINOS	x	x
	MOSEK	x	x
	PATHNLP		x
	SNOPT	x	x
MINLP	AlphaECP		x
	ANTIGONE		x
	BARON	x	x
	Bonmin	x	x
	Couenne	x	x
	DICOPT		x
	FilMINT	x	
	Knitro	x	x
	LINDOGlobal		x
	MINLP	x	
	SBB		x
	scip	x	x
MPCC	filterMPEC	x	
	Knitro	x	
	MILES		x
	NLPEC		x
	PATH	x	x
GDP	DE		x
	JAMS		x
Global	ANTIGONE		x
	ASA	x	
	BARON	x	x
	Couenne	x	x
	icos	x	
	LINDOGlobal		x
	PGAPack	x	
	PSwarm	x	
	scip	x	x

Table 7.1: List of solvers available on the NEOS Server. Data from [55].

Chapter 8

Numerical Results

In this chapter, we examine the results obtained from numerically solving the optimization problems introduced in Chapter 6, using the framework described in Chapter 7.
First, we summarize the previously introduced optimization problems and the parameters that need to be fixed in order to completely describe them. Then, we define representative test scenarios, that demonstrate the characteristic properties of the optimization problems. We examine the results obtained for solving these test scenarios with the different optimization problems and a variety of solvers. In some cases, we observe difficulties in finding optimal or even feasible solutions and introduce strategies for overcoming them.
Both the implementations of the discussed optimization problems, corresponding test scenarios and solution strategies and the obtained results can be downloaded at `https://github.com/ritterjul/opt-distillation`.

8.1 Optimization problems and corresponding test scenarios

In Chapter 6 we have formulated a number of optimization problems that formalize the problem of finding the optimal distillation sequence for a given separation task:
For the **superstructure approach**, we can choose between two models for distillation:

- "rigorous": Rigorous model derived from the MESH equations with non-ideal vapour-liquid equilibrium model, see Section 6.1.1.1
- "CMO": Simplified model using the assumption of constant molar overflow and constant relative volatilities, see Section 6.1.1.2.

Further, we have derived three formulations that reduce the feasible set to the column sequences corresponding to the three basic configurations, therefore we can choose between:

- "NLP": no additional constraints
- "MINLP": additional binary variables and linear constraints, see Section 6.1.2.1
- "MPCC": additional complementarity constraints, see Section 6.1.2.2
- "GDP": additional Boolean variables and disjunctive constraints, see Section 6.1.2.3.

This yields a total of eight optimization problems, that have different characteristics in terms of the type of optimization problem, the involved variables and the structure of the constraints.
In all cases, the number of stages in each distillation column ($N_s[i]\ , i \in I$), the entry stages of the feed stream (n_F) and interconnecting streams ($n_{CC},\ n_{CR}$) and the exit stage of the side-draw stream (n_{PS}) are parameters, that need to be fixed in order to completely describe the optimization problem. We define the following three test scenarios:

- "normal": $N_S[i] := 15\ , i \in I\ , n_F := 8\ , n_{CC} := 11\ , n_{CR} := 5\ , n_{PS} := 8$
- "large": $N_S[i] := 30\ , i \in I\ , n_F := 15\ , n_{CC} := 20\ , n_{CR} := 10\ , n_{PS} := 15$
- "x-large": $N_S[i] := 50\ , i \in I\ , n_F := 25\ , n_{CC} := 35\ , n_{CR} := 15\ , n_{PS} := 25.$

The first two scenarios represent the range of column sizes used in flowsheet-simulations for real-world applications. The third scenario acts as comparison to the Underwood shortcut model, which assumes infinitely many stages. We have distributed the entry stages of the interconnecting streams and the the exit stage of side-draw stream along the column.
Note, that the problem size (numbers of variables and constraints) increases linearly with $N_S[i]\ , i \in I$. The maximal value, such that the resulting optimization problems do not violate the size limitations of the demo version of the AMPL IDE, is given by the scenario "normal". Therefore, in some examinations, we will exclude (one of) the other two scenarios, since the resulting problems have to be submitted to the NEOS Server directly via the web interface, which takes more effort.
Additionally, we need to fix the parameter β, which indicates the quality of separation. We consider the three test scenarios $\beta = 0.95$, $\beta = 0.99$ and $\beta = 0.995$, which enforce product purities that are acceptable in most real-world applications.
For the **matrix-based approach**, we can choose between the enumeration-based optimization problem, see Section 6.2.1.1, or the simultaneous optimization problem, see Section 6.2.1.2.
Both use the Underwood model for distillation, which assumes that each distillation column has infinite stages and the products obtained are completely pure. Therefore, there are no additional parameters to be fixed.
Of course, in **both approaches**, in order to completely define a test scenario, we also need to fix the properties of the feed that is to be separated. They are described by the total mass flow F^{all}, the composition x^F, and most importantly the properties of the components present. We discuss the latter in the following section in detail:

8.1.1 Test scenarios of mixtures

The properties of a mixture are described by the parameters for the vapour pressures, activity coefficients and enthalpies, in the superstructure approach with "rigorous" model; or by the relative volatilities, in the superstructure approach with "CMO" model and the matrix-based approach.
We have received the parameters for the vapour pressures, activity coefficients and enthalpies from a BASF database, for the three examples of mixtures discussed in [48].
There, the authors considered feed mixtures of five components under the assumption of constant relative volatilities. They give values for the relative volatilities and the pressure, at which the former are valid, which is necessary to relate to the "rigorous" model (Note, that we need to convert the

pressure from atm to Pa by multiplying with the factor 101325). They also note the mass flow and composition of the feed. This data is presented in Table 8.1.

	"A"			"B"			"C"		
	P [atm]	0.3		P [atm]	1.0		P [atm]	2.0	
	F^{all}	100		F^{all}	200		F^{all}	200	
c	name of c	α_c	x_c^F	name of c	α_c	x_c^F	name of c	α_c	x_c^F
1	Benzene	10.50	0.04	Ethanol	4.10	0.2	Hexane	8.90	0.1
2	Toluene	4.04	0.06	Isopropanol	3.60	0.3	Heptane	5.70	0.2
3	Ethylbenzene	1.76	0.50	1-Propanol	2.10	0.2	Octane	3.20	0.3
4	Styrene	1.31	0.35	Isobutanol	1.42	0.2	Nonane	1.55	0.2
5	Methylstyrene	1.00	0.05	1-Butanol	1.00	0.1	Decane	1.00	0.2

Table 8.1: Pressure, mass flow and composition of the feed and relative volatilities for the three considered examples. Data from [48].

The data obtained from the BASF database is confidential and can not be presented here. Together, the data acquired fully describes three five component mixtures in both of the considered models. Since we want to examine feed mixtures of three components, we need to consider submixtures of the presented five component mixtures. There are a total number of $3\binom{5}{3} = 30$ possible submixtures. To pick representative examples, we do some preliminary tests:
We compare the vapour-liquid equilibrium obtained from using either the non-ideal model (2.9) - (2.11) or the constant relative volatility model (2.7) for the 30 three component submixtures available. Let $y(x)$ be the composition of vapour in equilibrium with liquid of given composition x. Similarly, let $x(y)$ bet the composition of liquid in equilibrium with vapour of given composition y.
For the non-ideal model, these functions are given implicitly by solving the system of equations:

$$\begin{aligned} \gamma_c(x,T)\ P_c^0(T)\ x_c - P\ y_c^{rig}(x) = 0, \quad c \in C \\ \sum_{c=1}^{N_C} y_c - 1 = 0 \end{aligned}$$

for $y^{rig}(x)$ and T; and analogously solving

$$\begin{aligned} \gamma_c(x^{rig}(y),T)\ P_c^0(T)\ x_c^{rig}(y) - P\ y_c = 0, \quad c \in C \\ \sum_{c=1}^{N_C} x_c^{rig}(y) - 1 = 0 \end{aligned}$$

for $x^{rig}(y)$ and T.
For the constant relative volatility model the functions are given explicitly:

$$\begin{aligned} y_c^{crv}(x) &= \frac{\alpha_c x_c}{\sum_{d=1}^{N_C} \alpha_d x_d}, \quad c \in C \\ x_c^{crv}(y) &= \frac{\frac{1}{\alpha_c} y_c}{\sum_{d=1}^{N_C} \frac{1}{\alpha_d} y_d}, \quad c \in C \end{aligned}$$

To compare the models described by the available data, we define two distance measures:

First, we consider the equally distributed points $f^{j,k} := (j \cdot 0.1, k \cdot 0.1, 1-(j+k)\cdot 0.1)^T$, $j = 1, \ldots, 8,\ k = 1, \ldots, 9-j$. We compare the corresponding equilibrium points by defining:

$$D_{points} := \sum_{j=1}^{8} \sum_{k=1}^{9-j} \sum_{c \in C} \left[\left(y_c^{rig}(f^{j,k}) - y_c^{crv}(f^{j,k}) \right)^2 + \left(x_c^{rig}(f^{j,k}) - x_c^{crv}(f^{j,k}) \right)^2 \right]$$

Additionally, we compare the corresponding equilibrium curves, that are calculated by iterating the equilibrium calculation: We define $y^{1,*}(f^{j,k}) := y^*(f^{j,k})$ and $y^{i+1,*}(f^{j,k}) := y^*(y^{i,*}(f^{j,k})$ with $* \in \{rig, crv\}$ and analogously for x. Then, we set:

$$D_{curves} := \sum_{j=1}^{3} \sum_{k=1}^{4-j} \sum_{i=1}^{5} \sum_{c \in C} \left[\left(y_c^{i,rig}(f^{2j,2k}) - y_c^{i,crv}(f^{2j,2k}) \right)^2 + \left(x_c^{i,rig}(f^{2j,2k}) - x_c^{i,crv}(f^{2j,2k}) \right)^2 \right]$$

Note, that we only choose even numbers of j, k for this measure, to obtain a similar number of summands for both measures.
We calculated the values of both measures for all 30 available submixtures with MATLAB and present the results in Table 8.2.

	"A"		"B"		"C"	
submixture	D_{points}	D_{curves}	D_{points}	D_{curves}	D_{points}	D_{curves}
123	**0.019**	**0.015**	0.014	0.055	0.665	1.230
124	0.021	0.011	0.040	0.143	0.438	0.930
125	0.751	0.164	0.027	0.038	0.544	0.933
134	0.017	0.004	0.022	0.058	0.515	0.304
135	0.469	0.578	0.018	0.025	0.552	0.245
145	0.457	1.702	0.025	0.028	0.342	0.222
234	**0.004**	**0.002**	**0.007**	**0.033**	0.068	0.139
235	0.627	0.633	**0.011**	**0.014**	**0.059**	**0.095**
245	0.616	1.679	0.005	0.009	0.066	0.150
345	0.527	2.063	0.374	0.177	**5.951**	**2.570**

Table 8.2: Comparison of the non-ideal and constant relative volatility vapour-liquid equilibrium models for the submixtures obtained from the examples given in Table 8.1. Mixtures chosen for further testing (bold), mixtures with "hard" separation (gray).

Note, that the Underwood shortcut method relies on finding the roots of a discontinuous function with poles at the constant relative volatilities α_c, as shown in Section 2.2.2.4. The closer the values α_c are, the steeper the function is in between the poles. Numerical experiments have shown that for mixtures with $\alpha_{c,d} = \frac{\alpha_c}{\alpha_d} < 1.5$ for some $c, d \in C$, using the Underwood equations as constraints for optimization is not stable. We say that mixtures with this property require a "hard" separation and mark them gray in Table 8.2.
We choose representative submixtures as test scenarios, by varying the following two properties: Agreement of the non-ideal and constant relative volatility vapour-liquid equilibrium models (given by the measures D_{points} and D_{curves}) and difficulty of the separation (given by the above heuristic).

We choose the following six test scenarios:

- A123: good agreement of models, easy separation
- A234: very good agreement of models, hard separation
- B235: good agreement of models, easy separation
- B245: very good agreement of models, hard separation
- C235: okay agreement of models, easy separation
- C345: very bad agreement of models, easy separation

The corresponding entries in Table 8.2 are marked bold. For three of these mixtures, the calculated equilibrium points and curves for both models are pictured in Appendix E in Figure E.1.
Note, that we could also fit the relative volatilities to the rigorous model by minimizing the defined measures D_{points} and D_{curves}. This could be a useful approach, if values for the relative volatilities of a mixture are not available. For the submixtures selected, however, we do not obtain a noteworthy improvement on the values presented in Table 8.2. This also shows, by the example of "C345", that some mixtures can not be described well by the constant relative volatility model.

8.2 Superstructure approach

In the previous section, we have summarized the eight optimization problems arising from the superstructure approach. There are 54 test scenarios: We have six choices in the mixtures and fix F^{all} to the corresponding value given in Table 8.1 and x^F to $(0.3, 0.4, 0.3)^T$. Further, we have three choices each in the column size and the demanded product purity.
We now describe our experiences with numerically solving these test scenarios with the introduced optimization problems and discuss the obtained results.
Note, that in the following we only use the implementation in the AMPL language, since the corresponding IDE has much fewer limitations on the problem size. This allows us to submit more jobs to the NEOS Server via the Kestrel interface, which is the fastest means of running multiple optimizations with different settings. Note though, that this excludes the "GDP" optimization problem, since there is no solver for GDP problems with AMPL input available on the NEOS Server.

8.2.1 Preliminary tests of solvers

Because of the large number of solvers available on the NEOS Server, we consider some preliminary tests to preselect good solvers for the optimization problems at hand.

8.2.1.1 NLP solvers

If we consider the superstructure formulation as an NLP without additional constraints for reducing the feasible set to the column sequences corresponding to basic configurations ("NLP"), we only need to consider two optimization problems that are characterized by the constraints describing the physical processes that govern distillation ("rigorous" and "CMO").

In this case, to test the performance of different solvers, it is sufficient to consider a single distillation column, which severely reduces the number of involved parameters. We obtain the corresponding formulations by simply deleting variables and constraints arising from the second column and interconnecting streams.
Further, we reduce the complexity of test scenarios, by only considering the column sizes "normal" and "large". The resulting problems can be solved within the AMPL IDE via the Kestrel interface.
We replace the constraint on the overall quality of separation (6.9) by constraints that enforce specifically one of the two possible splits $1|23$ and $12|3$, with corresponding purity parameter β. For example, for the former:

$$PC[1]y[1]_1^{(N_S)} \geq \beta F^{(all)} x_1^F$$
$$PR[1]x[1]_2^{(1)} \geq \beta F^{(all)} x_2^F$$

We only consider the two values of the purity parameter $\beta = 0.95$ and $\beta = 0.99$.
First, we fix the mixture to A123 and run the resulting eight test scenarios (two sizes, two splits, two purities) for the two optimization problems with all solvers for solving NLPs on the NEOS Server with AMPL input, listed in Table 7.1.
Some of the solvers execute with an error, if we do not specify (feasible) initial values. Therefore, we calculate feasible values by fixing $PC[1] := 0.5F^{all}$ and minimizing the maximum constraint violation. Here, we use the `Knitro` solver, which includes a pre-solve strategy that yields a feasible solution in all cases. We save the calculated solutions for the two models and two sizes and use them as initial values for every solver, to make the results even more comparable.
Despite this initialization strategy, two of the solvers executed with errors (`MOSEK`, `Ipopt`) for all test scenarios. The results for the remaining solvers are given in Table 8.3. If the solver stops at an optimal solution, we note the number of iterations executed, else, we note the termination message returned by the solver.
We select the four solvers with good performance (`CONOPT`, `filter`, `Knitro`, `LOQO`) for further benchmarking, in which we run the same test scenarios for the remaining five mixtures. The results can be found in Appendix E in Table E.1. Note that we have excluded the results for the solver `LOQO`, since it reaches the iteration limit (without obtaining a feasible solution) in almost all cases. The `CONOPT` solver performs best, while the `filter` and `Knitro` solvers obtain the optimal solution in the majority of cases, but also often fail with an infeasible solution.
We observe, that some mixtures lead to more numerical failures (infeasible solution/iteration limit) than others. As reasoned previously, the Underwood model is numerically unstable for "hard" separations ($\alpha_{c,d} < 1.5$ for some $c, d \in C$). This behaviour can also be observed for the "rigorous" and "CMO" models. Specifically, enforcing the split between the hard-to-separate components often leads to numerical failure. Note though, that in some of these cases (specifically, if none of the solvers obtain an optimal solution), there might not be a feasible solution, because the product purity specified can not be attained in the limited number of equilibrium stages. This is also indicated by the fact, that when increasing the number of stages more solvers return an optimal solution, even though the complexity of the problem increases.
In many of the following examinations, we will restrict ourselves to the mixture A123, which caused the smallest number of numerical failures.

		CONOPT	**filter**	**Knitro**	LANCELOT	**LOQO**	MINOS	SNOPT
"rigorous"	β							
"normal"								
Split 1	0.95	13	28	62	it limit	20	510	it limit
	0.99	12	128	11	it limit	20	infeas	181
Split 2	0.95	15	38	32	it limit	25	infeas	2898
	0.99	12	424	9	it limit	20	infeas	59
"large"								
Split 1	0.95	17	41	65	it limit	500	infeas	373
	0.99	11	23	16	it limit	463	infeas	131
Split 2	0.95	16	52	78	it limit	26	infeas	infeas
	0.99	9	23	19	it limit	170	infeas	603
"CMO"								
"normal"								
Split 1	0.95	21	11	150	137	20	infeas	138
	0.99	11	7	92	202	15	infeas	16
Split 2	0.95	15	11	32	267	24	infeas	231
	0.99	12	8	350	it limit	16	infeas	50
"large"								
Split 1	0.95	13	14	infeas	it limit	33	infeas	infeas
	0.99	11	7	infeas	229	19	infeas	238
Split 2	0.95	16	16	70	302	24	infeas	355
	0.99	8	7	53	358	16	infeas	40

Table 8.3: Results for eight test scenarios for the two single column optimization problems. Number of iterations (or termination message, in case of non-optimality) for different solvers on the NEOS Server. Selected solvers for further benchmarking (bold).

8.2.1.2 MINLP solvers

Of the six solvers for solving MINLPs on the NEOS Server with AMPL input, listed in Table 7.1, one is not available to access via the Kestrel interface (BARON), two execute with errors (Couenne, FilMINT) and the remaining three (Bonmin, Knitro, MINLP) execute successfully for a simple test example.

8.2.1.3 MPCC solvers

The number of solvers for solving MPCCs on the NEOS Server with AMPL input is already very small. In fact, of the three solvers listed in Table 7.1, filterMPEC is not available to access via the Kestrel interface and PATH returned an undocumented return code and no solution. Therefore, our examination restricts to the Knitro solver.

8.2.1.4 Global solvers

Of the seven solvers for solving optimization problems to globally optimality on the NEOS Server with AMPL input, listed in Table 7.1, ASA, PGAPack and PSwarm only handle bound- or linear constraints. Further, icos does not yet support the use of data files; we do not take on the extra effort to rewrite our implementation for this specific requirement.

Of the remaining ones, BARON and scip are not available to execute via the Kestrel interface and Couenne executes with an error. Therefore, our only possibility is to submit the jobs via the web interface to any of these three solvers, where they execute without errors for a simple test example.

8.2.2 Results

We now consider the six original optimization problems, arising from two choices in distillation model ("rigorous", "CMO") and three choices in additional constraints ("NLP", "MINLP", "MPCC").
In the last section we have motivated to choose the solvers CONOPT, filter and Knitro for the "NLP" case, Bonmin, Knitro and MINLP for the "MINLP" case and Knitro for the "MPCC" case.
For most of the following examinations, we restrict the test scenarios to the mixture A123. Further, we mostly consider the two column sizes "normal" and "large" and the three values of the purity parameter $\beta = 0.95$, $\beta = 0.99$, $\beta = 0.995$.
tho

8.2.2.1 Wiut any additional strategy

Handing the test scenarios for the above optimization problems directly over to any of the solvers will not give an optimal, or even feasible, solution: For example, submitting the "NLP" problem with "rigorous" model to CONOPT or filter fails instantly (for all test scenarios), because some of the expressions in the model can not be evaluated at the initial values. In other cases, the solver executes some iterations, but stops at an infeasible solution.
This indicates, that in order to obtain an optimal solution, we have to start with feasible initial values.

8.2.2.2 With initial values from distributing the mass flow

Our first approach is to calculate initial values, where all product and interconnecting streams have about equal mass flows (analogous to the single column case, where we distributed the feed mass flow equally to the two product streams).
To calculate them, we choose the "NLP" problem, fix the mass flows of the streams as shown in Figure 8.1 and minimize the maximum constraint violation. Again, we use the Knitro solver, which includes a pre-solve strategy that yields a feasible solution in all cases. We save the calculated solutions for the two models and three sizes and use them as initial values for every solver, to make the results even more comparable.

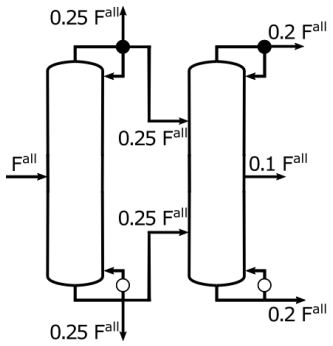


Figure 8.1: Specifications on the mass flows within the superstructure for calculating initial values.

Note, that for the "MINLP" and "MPCC" problems this is not a feasible solution, since it violates the constraints that restrict the feasible set to the column sequences corresponding to basic configurations. However, it satisfies the constraints describing the physical processes that govern distillation, which is already valuable.
In Table 8.4 we present the results using this initialization strategy. If an optimal solution was returned, we note the corresponding objective value and column sequence. Otherwise, we note the

termination message.

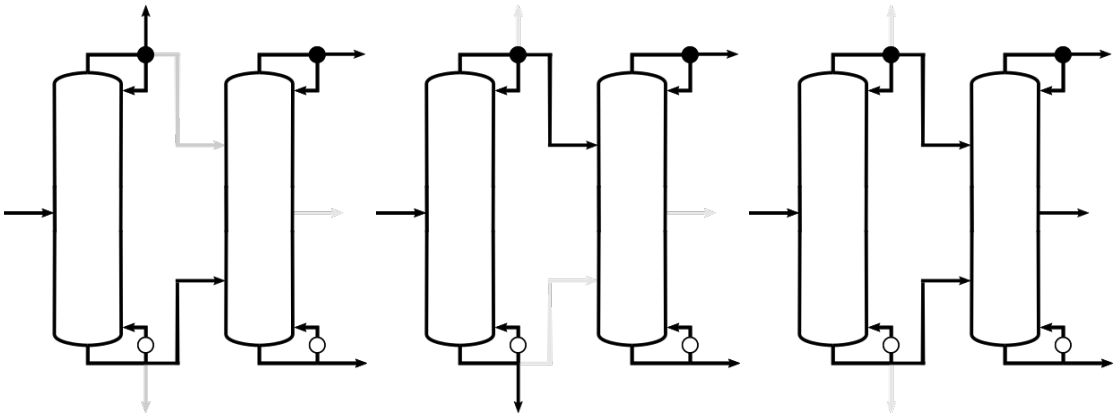
		"NLP"			"MINLP"			"MPCC"
		CONOPT	filter	Knitro	Bonmin	Knitro	MINLP	Knitro
"rigorous"	β							
"normal"								
	0.95	infeas	400.74 s.indirect	400.74 s.indirect	236.43 direct	infeas	231.55 indirect	infeas
	0.99	infeas	527.22 indirect	527.22 indirect	infeas	infeas	527.22 indirect	infeas
	0.995	infeas	1480.94 indirect	infeas	infeas	infeas	1480.94 indirect	infeas
"large"								
	0.95	infeas	185.37 s.direct	infeas	160.97 direct	154.49 prefrac	160.97 direct	154.49 prefrac
	0.99	235.59 s. indirect	439.98 s.direct	386.49 prefrac	infeas	infeas	infeas	386.49 prefrac
	0.995	infeas	4475.5 s.direct	279.61 s. indirect	infeas	infeas	infeas	infeas
"CMO"								
"normal"								
	0.95	infeas	260.99 indirect	infeas	error	infeas	infeas	infeas
	0.99	infeas	infeas	infeas	error	infeas	697.16 direct	infeas
	0.995	2395.5 indirect	infeas	infeas	error	infeas	infeas	infeas
"large"								
	0.95	194.49 s.direct	infeas	infeas	error	infeas	infeas	infeas
	0.99	450.83 s.direct	infeas	infeas	error	infeas	infeas	infeas
	0.995	infeas	infeas	infeas	error	infeas	289.35 indirect	infeas

Table 8.4: Results for six test scenarios for the six superstructure optimization problems using initial values from distributing the mass flow. Objective value (or termination message, in case of non-optimality) and sequence for different solvers on the NEOS Server.

We see, that this solution strategy yields some (locally) optimal solutions, but not reliably. It seems that, unlike for the single distillation column, initializing with feasible values is not sufficient to guarantee an optimal solution.

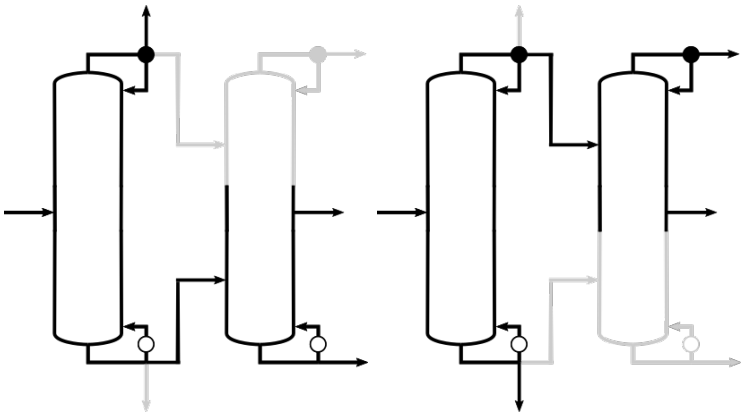
Note, that even though we formulated the "CMO" model as a simplification to eliminate the highly non-linear vapour pressure, activity coefficient and enthalpy equations, this model seams to be even harder to handle numerically than the "rigorous" model. Specifically, Bonmin executes with an error and Knitro stops at an infeasible solution for all test scenarios examined. Therefore, we focus on our original "rigorous" model in the following and only give results obtained from the "CMO" model with the solvers CONOPT, filter or MINLP, if appropriate.

Further, we note that even though we do not add additional constraints to restrict the feasible set to the column sequences corresponding to basic configurations in the "NLP" problem, it yields these sequences, or once that resemble them, as optimal solutions. This motivates, that we continue to include this problem formulation in the following.
In Figure 8.2 we picture the three column sequences corresponding to basic configurations as solutions of the superstructure formulation, with the (near-) zero streams marked gray. Analogously, we introduce the column sequences that resemble them in Figure 8.3.



(a) Direct sequence (b) Indirect sequence (c) Prefractionator

Figure 8.2: Column sequences corresponding to basic configurations, as solutions of the superstructure formulation. Near-zero streams (gray).



(a) Short direct sequence. (b) Short indirect sequence.

Figure 8.3: Additional column sequences that resemble the sequences in Figure 8.2, as solutions of the superstructure formulation. Near-zero streams (gray).

8.2.2.3 With increasing purity parameter

We enhance the initialization strategy from the last section to obtain optimal solutions more reliably: We observe, that when we reduce the purity parameter to $\beta = 0.8$, we obtain an optimal solution in almost all cases (using the initial values from the last section). However, this gives solutions that are not acceptable in the purity of the products. Therefore, we try to approach our desired solutions by gradually increasing β and taking the calculated solution as initial solutions for the next step. We choose the values $0.8, 0.9, 0.95, 0.99, 0.995$. The results for this strategy are given in Table 8.5.

	"NLP"			"MINLP"			"MPCC"
	CONOPT	filter	Knitro	Bonmin	Knitro	MINLP	Knitro
"rigorous" β							
"normal"							
0.8	166.86	141.63	141.63	147.14	147.14	159.09	141.62
	$\sim$ s. indirect	prefrac	prefrac	direct	direct	indirect	prefrac
0.9	234.16	234.16	234.16	185.33	185.33	192.08	723.48
	s. indirect	s. indirect	s. indirect	direct	direct	indirect	prefrac
0.95	400.74	400.74	400.74	236.43	236.43	231.55	infeas
	s. indirect	s. indirect	s. indirect	direct	direct	indirect	
0.99	527.22	527.22	infeas	647.96	647.96	527.22	infeas
	indirect	indirect		direct	direct	indirect	
0.995	1480.94	1480.94	1480.94	2094.5	2094.5	1480.94	infeas
	indirect	indirect	indirect	direct	direct	indirect	
"large"							
0.8	infeas	122.45	99.14	128.97	140.71	133.00	99.95
		mixed	$\sim$ prefrac	direct	indirect	direct	prefrac
0.9	153.81	153.81	127.6	151.78	166.45	151.78	127.5
	$\sim$ s. direct	$\sim$ s. direct	prefrac	direct	indirect	direct	prefrac
0.95	185.37	185.37	154.49	160.97	178.08	160.97	154.49
	s. direct	s. direct	prefrac	direct	indirect	direct	prefrac
0.99	439.98	439.98	235.59	181.41	192.45	181.41	386.49
	s. direct	s. direct	s. indirect	direct	indirect	direct	prefrac
0.995	error	infeas	279.61	196.41	200.35	196.41	infeas
			s. indirect	direct	indirect	direct	

Table 8.5: Results for 10 test scenarios for the three superstructure optimization problems (only "rigorous") using initial values from distributing the mass flow and increasing purity parameter. Objective value (or termination message, in case of non-optimality) and sequence for different solvers on the NEOS Server.

Again, if an optimal solution was returned, we note the corresponding objective value and column sequence, else, we note the termination message.
Note, that we extend the classification of column sequences introduced in Figures 8.2 and 8.3, by prefacing a sequence with $\sim$, if the characteristic streams are not necessarily (near-) zero, but significantly smaller than the others.
We observe, that the success rate of obtaining a (locally) optimal solution improves significantly. Therefore, we have developed a strategy for obtaining an optimal solution reliably.
However, note that we certainly do not obtain the globally optimal solution in most cases. This is to be expected, since the solvers tested, in general, calculate a locally optimal solution. However, this is not satisfying for the problem we want to solve. We try out some strategies for obtaining (an approximation of) the globally optimal solution in the following.

8.2.2.4 Enumeration by fixing sequence

The simplest strategy for approximating the globally optimal solution is to enumerate all column sequences, calculate the minimal objective for each sequence, and choose the column sequence with smallest objective value.

We enumerate the three column sequences corresponding to basic configurations, pictured in Figure 8.2, by fixing the appropriate streams to zero.
For each column sequence, we calculate initial values by minimizing the maximum constraint violation of the "NLP" problem. Again, we use the `Knitro` solver, which includes a pre-solve strategy that yields a feasible solution in all cases. We save the calculated solutions for the two models and three sizes and use them as initial values for every solver, to make the results even more comparable.
Now, we use the strategy introduced in the last section and gradually increase the purity parameter β, starting with the calculated initial solution.
Since we already fix the feasible set to a specific column sequence, it suffices to consider the "NLP" problem. The results obtained for the `Knitro` solver are presented in Table 8.6. The `filter` solver yields exactly the same results, while the `CONOPT` solver performs a bit worse, the corresponding results are given in Appendix E in Table E.2. We mark the obtained approximation of the globally optimal solution for each test scenario bold in both tables.

		direct	indirect	prefrac
"rigorous"	β			
"normal"				
	0.8	147.14	159.09	**141.62**
	0.9	**185.33**	192.08	723.48
	0.95	236.43	**231.55**	infeas
	0.99	647.96	**527.22**	infeas
	0.995	2094.5	**1480.94**	infeas
"large"				
	0.8	128.97	140.71	**99.95**
	0.9	151.78	166.45	**127.5**
	0.95	160.97	178.01	**154.49**
	0.99	**181.41**	192.45	386.49
	0.995	**196.41**	200.35	infeas
"xlarge"				
	0.8	125.58	139.14	**93.85**
	0.9	147.15	163.02	**112.14**
	0.95	156.73	174.93	**126.8**
	0.99	163.47	183.56	**144.16**
	0.995	165.22	184.9	**155.06**

Table 8.6: Results for 15 test scenarios for the single superstructure optimization problem (only "rigorous", "NLP") using fixed sequences, corresponding initial values and increasing purity parameter. Objective value (or termination message, in case of non-optimality) for `Knitro` solver. Approximation of the globally optimal solution (bold).

Note, that we obtain an optimal solution in almost all cases. Further note, that the large objective value for the case "normal", $\beta = 0.9$ (and similarly "large", $\beta = 0.99$) indicates, that the optimal solutions for higher values of β have very large objective values. This explains, why the solvers fail in exactly these cases.
Since the calculated solutions for each column sequence are again only locally optimal, we can not guarantee that we obtain the globally optimal solution with this strategy. However, numerical experiments show, that for many different initial values (for the same column sequence), we obtain the same optimal solution, which hints that it is globally optimal (for this column sequence).

We observe, that the optimal column sequence is very much dependent on the column size and value of the purity parameter. In fact, for our three column sizes, for the highest purity parameter value $\beta = 0.995$, we obtain three different optimal sequences. Note, that this behaviour can not be expressed by the Underwood model, which assumes infinite stages and perfectly pure products.

8.2.2.5 Enumeration by initial values

Another strategy for approximating the globally optimal solution is to vary the initial values.
We use the initial values corresponding to three column sequences, calculated in the last section, and run the optimization problems "NLP","MINLP" and "MPCC" (with "rigorous" model) for the three desired values of the purity parameter $\beta = 0.95$, $\beta = 0.99$ and $\beta = 0.995$. Note, that we always revert back to the initial values. This means, the computational effort is smaller than in the previous strategy, because we do not have to solve a series of optimization problems.
The results obtained for the `Knitro` solver are given in Table 8.7. For the remaining solvers, the results are presented in Appendix E in Table E.3.
We observe, that, in many cases, the solvers return (locally) optimal solutions. Further, in summary, we obtain all the solutions we have calculated by fixing the column sequence. Moreover, for some test scenarios where the previous approach stopped at an infeasible solution, this strategy yields a better, specifically feasible, solution.

	"NLP"			"MINLP"			"MPCC"		
	direct	indirect	prefrac	direct	indirect	prefrac	direct	indirect	prefrac
"rigorous" β									
"normal"									
0.8	143.32	159.09	141.63	147.14	159.09	141.62	147.14	159.09	141.63
	mix	indirect	prefrac	direct	indirect	prefrac	direct	indirect	prefrac
0.9	185.33	192.08	234.16	723.48	192.08	723.48	185.33	192.08	723.48
	direct	indirect	s.indirect	prefrac	indirect	prefrac	direct	indirect	prefrac
0.95	400.74	231.55	400.74	236.43	231.55	infeas	236.43	231.55	infeas
	s. indirect	indirect	s.indirect	direct	indirect		direct	indirect	
0.99	647.96	527.49	infeas	647.96	527.22	infeas	647.96	527.22	infeas
	direct	indirect		direct	indirect		direct	indirect	
0.995	2094.5	1480.94	infeas	2094.5	1480.94	infeas	infeas	1480.94	infeas
	direct	indirect		direct	indirect			indirect	
"large"									
0.8	116.37	139.86	99.14	128.97	140.71	99.95	128.97	140.71	infeas
	mix	$\sim$ indirect	$\sim$ prefrac	direct	indirect	prefrac	direct	indirect	
0.9	145.74	166.45	127.6	151.78	166.45	127.6	151.78	166.45	127.6
	mix	indirect	prefrac	direct	indirect	prefrac	direct	indirect	prefrac
0.95	160.41	178.09	154.49	160.97	178.08	154.49	160.97	178.01	infeas
	$\sim$ direct	indirect	prefrac	direct	indirect	prefrac	direct	indirect	
0.99	181.41	192.45	386.24	181.41	386.24	386.24	181.61	192.45	386.24
	direct	indirect	prefrac	direct	prefrac	prefrac	direct	indirect	prefrac
0.995	196.41	200.35	279.61	196.41	infeas	5525.9	infeas	200.35	infeas
	direct	indirect	s. indirect	direct		prefrac		indirect	

Table 8.7: Results for 10 test scenarios for the three superstructure optimization problem (only "rigorous") using different initial values. Objective value (or termination message, in case of non-optimality) and sequence for `Knitro` solver.

Note again, that in many cases the "NLP" problem returns one of the three column sequences corresponding to basic configurations, especially for higher values of the purity parameter β.
Of course, we can again combine this procedure with the strategy of gradually increasing β, to obtain an even better success rate for finding an optimal solution. However, this takes more computational effort, if only the solution for a single value of β is desired.

8.2.2.6 With global solvers

In the last two sections, we presented strategies for approximating the globally optimal solution by explicitly enumerating the alternative sequences. However, we have not obtained the unified strategy we had hoped for. Therefore, we lastly evaluate the option of using global solvers.
Of the three solvers available on the NEOS Server for NLPs, Couenne returns infeasible solutions and scip encounters a memory error in all of our test scenarios and optimization problems. Therefore, we can only examine the results obtained by the BARON solver.
However, for the "rigorous" model, for all our test scenarios the solver stops at an infeasible solution, even if we supply initial values calculated from distributing the mass flow or from specifying a column sequence. Luckily, for the "CMO" model, we do obtain feasible solutions in many cases.
In order to evaluate the results for this model, we first give an overview of the optimal objective values for each column sequence and test scenario in Table 8.8. By comparing with Table 8.6 we observe, that the objective values obtained from the "CMO" model are always higher than in the "rigorous" model, but in general exhibit the same trend. Therefore, this simplified model can also yield valuable insights.

		direct	indirect	prefrac
"CMO"	β			
"normal"				
	0.8	**155.61**	174.28	157.76
	0.9	**196.15**	212.00	
	0.95	**250.12**	260.99	
	0.99	697.16	**669.27**	
	0.995	2566.17	**2407.43**	
"large"				
	0.8	133.07	140.71	**99.95**
	0.9	159.00	181.19	**133.83**
	0.95	169.11		**161.99**
	0.99	**188.63**	212.36	430.62
	0.995	**202.29**	289.35	5700.83

Table 8.8: Results for 10 test scenarios for the single superstructure optimization problem (only "CMO","NLP") for fixed sequences. Objective values. Assumed globally optimal solutions (bold).

Note, however, that none of the strategies introduced in previous sections yield optimal solutions reliably for this model. We give detailed results in Appendix E: using the initial values calculated from distributing the mass flow, without and with increasing purity parameter, in Table E.4; for the enumeration by fixing the sequence, with corresponding initial values and increasing purity parameter, in Table E.5 and for the enumeration by initial values corresponding to different column sequences in Table E.6. We have combined them all to produce Table 8.8.

We now present the results obtained from solving the "NLP" and "MINLP" optimization problems with the BARON solver in Table 8.9, using the initial values calculated from distributing the mass flow in a previous section.

		"NLP"	"MINLP"
"CMO"	β		
"normal"			
	0.8	152.72	155.62
		mix	**direct**
	0.9	196.15	196.15
		direct	**direct**
	0.95	250.13	250.13
		direct	**direct**
	0.99	671.96	670.08
		indirect	**indirect**
	0.995	2407.43	infeas
		indirect	
"large"			
	0.8	123.6	133.07
		mix	direct
	0.9	153.52	159
		mix	direct
	0.95	168.48	169.11
		app. direct	direct
	0.99	188.63	212.36
		direct	indirect
	0.995	202.29	202.33
		direct	**direct**

Table 8.9: Results for 10 test scenarios for the two superstructure optimization problems (only "CMO", "NLP"/"MINLP") with initial values from distributing the mass flow. Objective values (or termination message, in case of non-optimality) and sequence for BARON solver. Sequences that align with the assumed globally optimal solution (bold).

We observe, that we obtain the same sequences as the (assumed to be) globally optimal solution in some cases, marked bold, but not consistently.

Note, that if we do not provide any initial values, we obtain almost identical results, which is an improvement compared to all previous strategies.

However, we note that this strategy is much more computationally expensive than all previous strategies. In the previous settings, the solver returned a solution (or a termination message) in less than 10 seconds, where as a call of the BARON solver with any of our test scenarios takes up to 10 minutes. This outweighs the additional effort that must be taken to calculate initial values or solve a series of optimization problems.

8.3 Matrix-based approach

We have previously introduced two optimization problems arising from the matrix-based approach. In the enumeration strategy, we need to solve a series of NLPs (one for every basic configuration), whereas in the simultaneous optimization approach, we need to solve a single MINLP.
There are six test scenarios given by the six choices in the mixtures. We fix F^{all} to the corresponding value given in Table 8.1 and x^F to $(0.3, 0.4, 0.3)^T$. In order to generate more test scenarios, we might also vary the feed composition x^F.
We now describe our experiences with numerically solving these test scenarios with the introduced optimization problems and discuss the obtained results.
Note, that in the following we, again, only use the implementation in the AMPL language.

8.3.1 Results

In contrast to the superstructure approach, the optimization problems arising from the matrix-based approach are easy to handle numerically:
Using the enumeration strategy, we obtain an optimal solution for every basic configuration within few iterations (< 10) without providing initial values or any further input, by using the `Knitro` solver. The results for all six mixtures are given in Table 8.10, with the (assumed to be) globally optimal solution marked bold.

	A123	A234	B235	B245	C235	C345
direct	172.02	341.78	478.89	583.36	390.92	547
indirect	198.66	361.77	536.18	625.55	450.28	594.32
prefrac	***136.12***	***304.73***	***362.15***	***519.49***	***324.73***	***449.09***

Table 8.10: Results for all six mixtures for the matrix-based enumeration strategy. Assumed globally optimal solution (bold). Solution obtained by the simultaneous optimization (italic). Both strategies using the `Knitro` solver.

Note, that even for the mixtures, where we expected numerical problems due to a "hard" separation, we do not observe any numerical failures.
Similarly, we obtain an optimal solution for the simultaneous optimization without providing initial values, by again using the `Knitro` solver. In all cases, the calculated solutions align with one of the solutions obtained in the enumeration strategy. We mark them italic in Table 8.10.
We observe, that this strategy obtains the (assumed to be) globally optimal solution in all cases. However, we note, that this solution is always attained for the prefractionator configuration. Therefore, these six test scenarios might not be representative.
We generate more representative test scenarios by fixing the mixture to A123, but varying the feed composition x^F. We let $(x^F)^{(j,k)} = (j \cdot 0.1, k \cdot 0.1, 1 - (j+k) \cdot 0.1)$, $j = 1, \ldots, 8$, $k = 1, \ldots, 9 - j$.
For these 36 test scenarios, we, again, run both strategies without providing initial values and using the `Knitro` solver. The results are given in Table 8.11, with the (assumed to be) globally optimal solution marked bold and the solution obtained by the simultaneous optimization marked italic.

x_1^F	x_2^F	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
0.1	direct	115.21	130.25	145.05	159.69	174.25	188.75	203.20	217.62
	indirect	105.84	124.36	142.18	159.59	176.75	193.73	210.58	227.34
	prefrac	***107.60***	***115.58***	***123.57***	***131.56***	***139.54***	***147.53***	***155.52***	***163.50***
0.2	direct	119.59	135.39	150.64	165.62	180.42	195.12	209.74	
	indirect	120.32	140.99	160.11	178.45	196.31	213.85	231.17	
	prefrac	***109.88***	***117.86***	***125.85***	***133.84***	***141.82***	***149.81***	***157.8***	
0.3	direct	124.96	141.27	156.83	172.02	187.00	201.84		
	indirect	137.59	159.71	179.70	198.66	217.01	234.96		
	prefrac	***112.16***	***120.15***	***128.13***	***136.12***	***144.1***	***152.09***		
0.4	direct	131.32	147.81	163.52	178.84	193.93			
	indirect	157.68	180.32	200.73	220.05	238.70			
	prefrac	***129.39***	***133.63***	***137.87***	***142.11***	***146.38***			
0.5	direct	***138.44***	**154.88**	*170.62*	185.99				
	indirect	179.94	202.42	222.93	242.39				
	prefrac	155.65	*159.89*	**164.13**	***168.37***				
0.6	direct	**146.05**	**162.33**	***178.04***					
	indirect	203.57	225.62	246.02					
	prefrac	*181.9*	*186.14*	190.38					
0.7	direct	**153.96**	**170.06**						
	indirect	228.06	249.58						
	prefrac	*208.16*	*212.4*						
0.8	direct	**162.05**							
	indirect	253.06							
	prefrac	*234.41*							

Table 8.11: Results for mixture A123 and test scenarios arising from varying the feed composition for the matrix-based enumeration strategy. Assumed globally optimal solution (bold). Solution obtained by the simultaneous optimization (italic). Both strategies using the `Knitro` solver.

Again, we obtain optimal solutions consistently and with few iterations. We note that for feed compositions with $x_1^F > 0.5$, the prefractionator is not always the optimal configuration. In these cases, the simultaneous optimization strategy does not reliably obtain the (assumed to be) globally optimal solution.

To improve this strategy, we try different approaches:

Firstly, we can use other solvers that are available for MINLPs on the NEOS Server. However, of the ones available to access via the Kestrel interface and executing without errors, only `Bonmin` is able to calculate a solution without feasible initial values. Further, it obtains the globally optimal solution in less cases then the `Knitro` solver. The complete results are given in Appendix E in Table E.7.

Secondly, we can use solvers intended for global optimization. However, none of the ones available on the NEOS Server, that can treat non-linear constraints (`BARON`, `Couenne`, `scip`), are able to treat the non-smooth constraint arising from the hybrid approach.

8.4 Summary

We summarize the results gathered in the previous sections:
The **superstructure approach** results in optimization problems that are difficult to handle numerically. Specifically, the "CMO" model yields feasible solutions for a very limited number of test scenarios and solvers, therefore we concentrate on the "rigorous" model. We need to introduce some strategies in order to obtain solutions:

- If we do not provide initial values, we do not obtain feasible solutions.
- If we provide initial values calculated from distributing the mass flow, we obtain feasible solutions in some cases, but not reliably.
- If we provide initial values calculated from distributing the mass flow and gradually increase the purity parameter, we obtain feasible solutions in most cases.

However, this will, in general, not yield (a good approximation of) the globally optimal solution. In order to improve the approximation, we can:

- Enumerate all column sequences by fixing appropriate streams.
- Enumerate all column sequences by using corresponding initial values.
- Use the global solver BARON.

The first two options yield (a good approximation of) the globally optimal solution reliably and can be combined with the strategies introduced above. However, they are hard to generalize for larger superstructures and their complexity grows with the number of configurations (exponentially).
The third option is the only unified approach, that does not involve solving a series of optimization problems. However, it only obtains solutions for the simplified model "CMO". Therefore the objective values differ from the calculation with the "rigorous" model, but in most cases the rank-list of sequences aligns. We note though, that this approach does not approximate the globally optimal solution reliably and further is much more computationally expensive (even compared to the series of optimization problems in the first two options).
We observe, that we often obtain a column sequence corresponding to a basic configuration as solution, even if we do not add any additional constraints that restrict to these. Therefore, it is sufficient to only consider NLPs, which are supported by most solvers.
In comparison, the **matrix-based approach** results in much smaller and more tractable optimization problems. We do not have to provide initial values or implement any other strategies in order to obtain feasible solutions. The enumeration strategy returns a rank-list of all basic configurations and the corresponding globally optimal solution reliably. The simultaneous optimization strategy also yields a feasible solution in all cases, but is not reliable in obtaining the globally optimal solution.
Note, that both strategies (and their implementations) are general in N_C and therefore easily extendable to problems with $N_C > 3$. However, the very simplifying assumptions of the Underwood model lead to results with limited information content. While the optimal configuration in the superstructure approach depends on parameters such as the column size and the desired purity of products, which are set or restricted in real-world applications, the matrix-based approach yields one optimal configuration.

Chapter 9

Conclusion and Outlook

In this thesis, we have examined methods for finding the optimal distillation system for a given separation task.

We have reviewed literature on this topic and have taken elements of multiple sources to develop two new approaches to the problem: a superstructure approach and a matrix-based approach. We have derived the optimization problems arising from both in detail and discussed the technical framework we used to numerically solve them. Lastly, we examined a variety of numerical results that demonstrated the advantages and disadvantages of both approaches.

Based on these results, we propose a **combined strategy** for larger ($N_C > 3$) separation tasks arising in real-world applications:

We could use the enumeration strategy of the matrix-based approach to draw a full rank-list of all basic configurations under simplifying assumptions. We do not have to provide any additional insight in order to obtain these results reliably. Based on this rank-list we could eliminate configurations that perform very poorly or are not acceptable because of other constraints (for example the level of integration, i.e. the number of separations integrated into one column).

For the remaining configurations, we could create a superstructure that encompasses them all and enables rigorous modelling. Further, we could set parameters such as column size or desired purity to values derived from our real-word constraints, or create multiple test scenarios of acceptable values. Then, we could use the strategies developed for the superstructure approach in order to calculate the optimal solution for each configuration and therefore (a good approximation of) the overall globally optimal solution.

Though the resulting strategy relies heavily on enumeration and can not be represented as a single optimization problem, it does provide a simplification for real-world applications:

Firstly, the pre-screening of all basic configurations based on the Underwood shortcut method reduces the number of considered configurations and is both computationally cheap and robust.

Secondly, the superstructure approach provides a simplified method for enumerating different configurations with rigorous models. In industry, where separation tasks are often part of complex flowsheet simulations, instead of creating a new flowsheet for each configuration, we can create a single one for the superstructure.

The implementation of this strategy and the test on real-world applications are topics that may be examined in the future.

Part IV

Appendix

Appendix A

Details on the Underwood method

The approach described follows the argumentation in [11].

We derive a relationship between the compositions of the liquids leaving adjacent stages in the rectifying section. By material balance around the rectifying section above stage $n \in \{n_f+1, \dots, N_S-1\}$, we obtain:

$$D\ x_c^D = V^{(r)}\ y_c^{(n)} - L^{(r)}\ x_c^{(n+1)}, \quad c = 1, \dots, N_C. \tag{A.1}$$

Let φ be a solution of the Underwood equation for the rectifying section (2.26), then it holds:

$$\begin{aligned}
\frac{1}{V^{(r)}} D(R+1) &= \frac{1}{V^{(r)}} \sum_{c=1}^{N_C} \frac{\alpha_c D x_c^D}{\alpha_c - \varphi} \\
&\overset{\text{(A.1)}}{=} \sum_{c=1}^{N_C} \frac{\alpha_c y_c^{(n)}}{\alpha_c - \varphi} - \frac{L^{(r)}}{V^{(r)}} \sum_{c=1}^{N_C} \frac{\alpha_c x_c^{(n+1)}}{\alpha_c - \varphi} \\
&= \sum_{c=1}^{N_C} \frac{\alpha_c \frac{\alpha_c x_c^{(n)}}{\sum_{d=1}^{N_C} \alpha_d x_d^{(n)}}}{\alpha_c - \varphi} - \frac{L^{(r)}}{V^{(r)}} \sum_{c=1}^{N_C} \frac{\alpha_c x_c^{(n+1)}}{\alpha_c - \varphi}
\end{aligned} \tag{A.2}$$

where we use the assumption of constant relative volatility in the last step. We define $E(x, \varphi) := \sum_{c=1}^{N_C} \frac{\alpha_c x_c}{\alpha_c - \varphi}$. Then by (2.26) it holds $E(x^D, \varphi) = R+1$ and further:

$$\begin{aligned}
\frac{R}{R+1} E(x^{(n+1)}, \varphi) &= \frac{L^{(r)}}{V^{(r)}} \sum_{c=1}^{N_C} \frac{\alpha_c x_c^{(n+1)}}{\alpha_c - \varphi} \\
&\overset{\text{(A.2)}}{=} \sum_{c=1}^{N_C} \frac{\alpha_c \frac{\alpha_c x_c^{(n)}}{\sum_{d=1}^{N_C} \alpha_d x_d^{(n)}}}{\alpha_c - \varphi} - \frac{1}{V^{(r)}} D(R+1) \\
&= \frac{\sum_{c=1}^{N_C} \frac{\alpha_c^2 x_c^{(n)}}{\alpha_c - \varphi}}{\sum_{c=1}^{N_C} \alpha_c x_c^{(n)}} - 1 \\
&= \frac{\sum_{c=1}^{N_C} \frac{\alpha_c^2 x_c^{(n)} - \alpha_c x_c^{(n)} (\alpha_c - \varphi)}{\alpha_c - \varphi}}{\sum_{c=1}^{N_C} \alpha_c x_c^{(n)}} \\
&= \frac{E(x^{(n)}, \varphi)\varphi}{\sum_{c=1}^{N_C} \alpha_c x_c^{(n)}}.
\end{aligned} \tag{A.3}$$

Analogously, we derive for a solution ψ of the Underwood equation for the stripping section (2.27):

$$\frac{S+1}{S}E(x^{(n+1)},\psi) = -\frac{E(x^{(n)},\psi)\varphi}{\sum_{c=1}^{N_C}\alpha_c.x_c^{(n)}}$$

We consider distinct solutions φ, φ' of the Underwood equation for the rectifying section (2.26). Then $E(x^D,\varphi) = R+1 = E(x^D,\varphi')$ and therefore:

$$\begin{aligned}
1 &= \frac{E(x^D\varphi)}{E(x^D,\varphi')} \\
&= \frac{E(x^{(N_S)},\varphi)}{E(x^{(N_S)},\varphi')} \\
&\overset{(A.3)}{=} \frac{\varphi}{\varphi'}\frac{E(x^{(N_S-1)},\varphi)}{E(x^{(N_S-1)},\varphi')} \\
&= (\frac{\varphi}{\varphi'})^{N_S-n}\frac{E(x^{(n)},\varphi)}{E(x^{(n)},\varphi')}
\end{aligned}$$

which finally gives

$$\frac{E(x^{(n)},\varphi)}{E(x^{(n)},\varphi')} = (\frac{\varphi'}{\varphi})^{N_S-n}.$$

For minimum reflux conditions and therefore infinite stages, there exists a so called *pinch point* where the composition does not change between stages. We denote the composition as x^P and observe:

$$\frac{R}{R+1} \overset{(A.3)}{=} \frac{\varphi}{\sum_{c=1}^{N_C}\alpha_c x_c^P}$$

with the solution

$$x_c^P = \frac{x_c^D\varphi}{R(\alpha_c-\varphi)}. \tag{A.4}$$

We denote by $x^{P,c}$ the pinch point composition obtained for the solution φ_c. Then for $d \neq c$:

$$\begin{aligned}
E(x^{P,c},\varphi_d) &\overset{(A.4)}{=} \sum_{e=1}^{N_C}\frac{\alpha_e x_e^D\varphi_c}{(\alpha_e-\varphi_d)R(\alpha_e-\varphi_c)} \\
&= \frac{1}{R(\varphi_d-\varphi_c)}\sum_{e=1}^{N_C}\frac{\alpha_e x_e^D(\varphi_d-\varphi_c)}{(\alpha_e-\varphi_d)(\alpha_e-\varphi_c)} \\
&= \frac{1}{R(\varphi_d-\varphi_c)}\sum_{e=1}^{N_C}(\frac{\alpha_e x_e^D}{\alpha_e-\varphi_d} - \frac{\alpha_e x_e^D}{\alpha_e-\varphi_c}) \\
&\overset{(2.26)}{=} \frac{1}{R(\varphi_d-\varphi_c)}((R+1)-(R+1)) \\
&= 0.
\end{aligned}$$

Furthermore, we observe

$$E(x^{P,c},\varphi_c) = \sum_{e=1}^{N_C}\frac{\alpha_e x_e^D\varphi_c}{R(\alpha_e-\varphi_c)^2} > 0.$$

First, we assume that all components distribute to the distillate product: $x_c^D > 0,\ c = 1, \ldots, N_C$. Then all solutions $\varphi_c,\ c = 1, \ldots, N_C$ exist. Since $\varphi_{N_C} < \varphi_{N_C-1} \ldots < \varphi_1$ this yields:

$$\frac{E(x^{(n)}, \varphi_c)}{E(x^{(n)}, \varphi_{N_C})} = (\frac{\varphi_{N_C}}{\varphi_c})^{N_S - n} \overset{N_S \to \infty}{\to} 0.$$

This implies $E(x^{(n)}, \varphi_c) = 0,\ c = 1, \ldots, N_C - 1$ and $E(x^{(n)}, \varphi_{N_C}) \neq 0$. Therefore, by calculating down the column, we will reach the pinch point composition x^{P,N_C}.

Now, we consider possible splits under the assumption that the separation is not perfect.

For the split $1 \cdots (N_C - 1)|N_C$, all components appear in the distillate product. Therefore, as derived above, we reach the pinch point composition x^{P,N_C} by calculating down the column. By inverting the equation and calculating up the column from the feed stage, we get the same result. This means, the pinch point is reached at the feed stage. We obtain $N_C - 1$ equations $E(x^{(n_F)}, \varphi_c) = 0,\ c = 1, \ldots, N_C - 1$ for the composition of the liquid leaving the feed stage.

For the split $1 \cdots (N_C - 2)|(N_C - 1)N_C$, the N_Cth component does not appear in the distillate. We reproduce the derivation from above with one less component and observe that by calculating down the column we reach the pinch point composition x^{P,N_C-1}. In order to reach this point for the upward calculation, we must have $E(x^{(n_F)}, \varphi_{N_C-1}) \neq 0$, else the expression would also be 0 for all stages above the feed stage. $\lightning$ Then n stages above the feed stage it holds:

$$\frac{E(x^{(n_F+n)}, \varphi_{N_C})}{E(x^{(n_F+n)}, \varphi_{N_C-1})} = (\frac{\varphi_{N_C}}{\varphi_{N_C-1}})^n \frac{E(x^{(n_F)}, \varphi_{N_C})}{E(x^{(n_F)}, \varphi_{N_C-1})} \overset{n \to \infty}{\to} 0.$$

This means, the equation $E(x, \varphi_{N_C}) = 0$ for the pinch point is already fulfilled by upwards calculation. To guarantee that the other equations hold, however, we need to require them for the feed stage. We obtain $N_C - 2$ equations $E(x^{(n_F)}, \varphi_c) = 0,\ c = 1, \ldots, N_C - 2$.

Now, let's consider a general separation with heavy key HK and light key LK. Extending the argumentation from above, we obtain $HK - 1$ equations $E(x^{(n_F)}, \varphi_c) = 0,\ c = 1, \ldots, HK - 1$. Analogous arguments hold for the stripping section (with ψ instead of φ), so that we obtain $N_C - LK$ equations $E(x^{(n_F)}, \psi_c) = 0,\ c = LK + 1, \ldots, N_C$.

In total, for sharp splits ($HK = LK + 1$), we obtain N_C equations for the composition of the liquid leaving the feed stage. But there are only $N_C - 1$ independent variables $x_c^{(n_F)},\ c = 1, \ldots, N_C - 1$, since the summation equation implies $x_{N_C}^{(n_F)} = 1 - \sum_{c=1}^{N_C-1} x_c^{(n_F)}$. Therefore, one of the equations must be dependent. By the definition of E, this implies $\varphi_c = \psi_d$ for some $c \in \{1, \ldots, HK - 1\},\ d \in \{LK + 1, \ldots, N_C\}$ and based on their bounds we conclude $\varphi_{LK} = \psi_{LK+1}$. Note, that for non-sharp splits we obtain even more equations and therefore multiple coinciding values [12].

Appendix B

Proof of the closed-form expression of the Catalan numbers

The series of Catalan numbers $C_i,\ i \in \mathbb{N}$ is uniquely defined by the initial values $C_0 = 1,\ C_1 = 1$ and the following recurrence relation:

$$C_i = \sum_{j=0}^{i-1} C_j C_{i-1-j}, \quad i \geq 2. \tag{B.1}$$

Note, that this implies the relation $S_{i+1} = C_i,\ i \in \mathbb{N}$ by induction:

Trivially, $S_1 = 1 = C_0$ and if the relation holds up to $i - 1 \in \mathbb{N}$, then:

$$S_{i+1} = \sum_{j=1}^{(i+1)-1} S_j S_{(i+1)-j} \overset{(4.1)}{=} \sum_{j=0}^{i-1} S_{j+1} S_{i-j} \overset{IH}{=} \sum_{j=0}^{i-1} C_j C_{i-1-j} \overset{(B.1)}{=} C_i.$$

We use the method of generating functions to derive a closed-form expression for C_i. For $x \in \mathbb{R}$, we define the power series $C(x) := \sum_{i=0}^{\infty} C_i x^i$. By the Cauchy -Hadamard theorem, the radius of convergence of C is given by

$$\frac{1}{R} = \limsup_{i \to \infty} (|C_i|^{\frac{1}{i}}).$$

In [57], the author motivates the ansatz

$$C_i \leq \frac{M^i}{i^2}, \quad \forall i \in \mathbb{N} \backslash \{0\}. \tag{B.2}$$

Again, we prove this by induction. Assume (B.2) holds up to $i \in \mathbb{N}$, then we obtain

$$\begin{aligned} C_{i+1} &= \sum_{j=0}^{(i+1)-1} C_i C_{(i+1)-1-j} \\ &= 2C_0 C_i + \sum_{j=1}^{i-1} C_j C_{i-j} \\ &\overset{IH}{\leq} 2\frac{M^i}{i^2} + \sum_{j=1}^{i-1} \frac{M^j}{j^2} \frac{M^{i-j}}{(i-j)^2} \end{aligned}$$

$$
\begin{aligned}
&= M^i \left(\frac{2}{i^2} + \sum_{j=1}^{i-1} \frac{1}{j^2(i-j)^2} \right) \\
&= M^i \left(\frac{2}{i^2} + \sum_{j=1}^{i-1} \left(\frac{1}{i}(\frac{1}{j} + \frac{1}{i-j}) \right)^2 \right) \\
&= M^i \frac{1}{i^2} \left(2 + \sum_{j=1}^{i-1} \left(\frac{1}{j^2} + \frac{2}{j(i-j)} + \frac{1}{(i-j)^2} \right) \right) \\
&= M^i \frac{1}{i^2} \left(2 + 2\sum_{j=1}^{i-1} \frac{1}{j^2} + 2\sum_{j=1}^{i-1} \frac{1}{j(i-j)} \right) \\
&= M^i \frac{2}{i^2} \left(1 + \sum_{j=1}^{i-1} \frac{1}{j^2} + \sum_{j=1}^{i-1} \left(\frac{1}{i}(\frac{1}{j} + \frac{1}{i-j}) \right) \right) \\
&= M^i \frac{2}{i^2} \left(1 + \sum_{j=1}^{i-1} \frac{1}{j^2} + \frac{2}{i} \sum_{j=1}^{i-1} \frac{1}{j} \right).
\end{aligned}
$$

We define $f(i) := \frac{2}{i^2}(1 + \sum_{j=1}^{i-1} \frac{1}{j^2} + \frac{2}{i} \sum_{j=1}^{i-1} \frac{1}{j})(i+1)^2$. If $f(i) \leq M$, then (B.2) holds for $i+1$. Note, that

$$
\lim_{i \to \infty} f(i) = \lim_{i \to \infty} 2(1 + \frac{1}{i})^2 (1 + \sum_{j=1}^{i-1} \frac{1}{j^2} + \frac{2}{i} \sum_{j=1}^{i-1} \frac{1}{j}) = 2(1 + \frac{\pi^2}{6}) \approx 5.289868
$$

since the asymptotic behaviour of the harmonic series is $\sum_{j=1}^{i-1} \frac{1}{j} = \gamma + \ln(i) + \mathcal{O}(\frac{1}{i})$. In our case, it will suffice to prove a weaker bound.
In order to prove $f(i) > f(i+1)$, since $(1 + \frac{1}{i})^2 > (1 + \frac{1}{i+1})^2$, it suffices to show:

$$
\begin{aligned}
& 1 + \sum_{j=1}^{i-1} \frac{1}{j^2} + \frac{2}{i} \sum_{j=1}^{i-1} \frac{1}{j} \geq 1 + \sum_{j=1}^{i} \frac{1}{j^2} + \frac{2}{i+1} \sum_{j=1}^{i} \frac{1}{j} \\
\Leftrightarrow \quad & \frac{2}{i} \sum_{j=1}^{i-1} \frac{1}{j} \geq \frac{1}{i^2} + \frac{2}{i+1} \sum_{j=1}^{i} \frac{1}{j} \\
\Leftrightarrow \quad & \sum_{j=1}^{i-1} \frac{1}{j} - (1 - \frac{1}{i+1}) \sum_{j=1}^{i} \frac{1}{j} \geq \frac{1}{2i} \\
\Leftrightarrow \quad & -\frac{1}{i} + \frac{1}{i+1} \sum_{j=1}^{i} \frac{1}{j} \geq \frac{1}{2i} \\
\Leftrightarrow \quad & \sum_{j=1}^{i} \frac{1}{j} \geq \frac{3(i+1)}{2i}.
\end{aligned}
$$

This inequality holds true for $i = 4$:

$$
1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} = \frac{50}{24} \geq \frac{45}{24} = \frac{3 \cdot 5}{2 \cdot 4}.
$$

Since the left side is monotonically increasing and the right side is monotonically decreasing in i, the inequality holds true for all $i \geq 4$. Therefore, $f(i) \leq f(5) = \frac{469}{50} < 10 =: M, \ \forall i \geq 5$.

Computing C_i for $i = 1, 2, 3, 4$ from the recurrence relation (B.1) shows, that the inequality (B.2) holds true in these cases. Therefore, using the derived bound on $f(i)$, we can conclude by induction that the inequality (B.2) holds true for all $i \in \mathbb{N}\backslash\{0\}$. This also implies the weaker inequality

$$C_i \leq M^i, \quad \forall i \in \mathbb{N}\backslash\{0\}$$

and therefore $R \geq \frac{1}{M} = \frac{1}{10} > 0$. In fact, it is possible to prove $R \geq \frac{1}{4}$ by deriving an inequality similar to (B.2) with exponent $\frac{3}{2}$ instead of 2.

Now, let $x \in]-\frac{1}{10}, \frac{1}{10}[$. Then we obtain

$$\begin{aligned} C(x) &= C_0 x^0 + \sum_{i=1}^{\infty} \sum_{j=0}^{i-1} C_j C_{i-1-j} x^i \\ &= 1 + \sum_{i=0}^{\infty} \sum_{j=0}^{i} C_j C_{i-j} x^{i+1} \\ &= 1 + x \sum_{i=0}^{\infty} \sum_{j=0}^{i} (C_j x^j)(C_{i-j} x^{i-j}) \\ &= 1 + x(\sum_{k=0}^{\infty} C_k x^k)(\sum_{l=0}^{\infty} C_l x^l) \\ &= 1 + xC(x)^2 \end{aligned}$$

where we can use the Cauchy product of infinite series in the second to last step, since we proved the (absolute) convergence of C. Then $C(x)$ satisfies for $x \neq 0$:

$$\begin{aligned} & 0 = C(x)^2 - \frac{1}{x} C(x) + \frac{1}{x} \\ \Leftrightarrow \quad & 0 = (C(x) - \frac{1}{2x})^2 - \frac{1}{4x^2} + \frac{1}{x} \\ \Leftrightarrow \quad & C(x) = \frac{1}{2x} \pm \sqrt{\frac{1}{4x^2} - \frac{1}{x}} = \frac{1 \pm \sqrt{1-4x}}{2x} =: C_{\pm}(x). \end{aligned}$$

We notice that $C(0) = C_0 = 1$ by definition. Since $\lim_{x \downarrow 0} C_+(x) = \infty$, we can eliminate this solution. By the rule of l'Hôspital

$$\lim_{x \to 0} C_-(x) = \lim_{x \to 0} \frac{-(-4)\frac{1}{2}\frac{1}{\sqrt{1-4x}}}{2} = \lim_{x \to 0} \frac{1}{\sqrt{1-4x}} = 1.$$

This means $C(x) = C_-(x)$ on the domain of convergence of C. Therefore the coefficients C_i of the power series C can be obtained as the coefficients of the Taylor series of the function C_- in 0. Note, that the Taylor series is unique.

First we calculate the Taylor series of $g(y) := \sqrt{1+y}$ in 0. Since $g^{(i)}(y) = \prod_{k=0}^{i-1}(\frac{1}{2} - k)(1+y)^{(\frac{1}{2}-i)}$, we obtain

$$\sqrt{1+y} = \sum_{i=0}^{\infty} \frac{1}{i!} g^{(i)}(0) y^i = \sum_{i=0}^{\infty} \frac{1}{i!} \prod_{k=0}^{i-1} (\frac{1}{2} - k) y^i$$

where the coefficients can be simplified to:

$$\begin{aligned}
\frac{1}{i!}\prod_{k=0}^{i-1}(\frac{1}{2}-k) &= \frac{1}{i!}\frac{1}{2^i}(-1)^{i-1}\prod_{k=1}^{i-1}(2k-1) \\
&= \frac{1}{i!}\frac{1}{2^i}(-1)^{i-1}\frac{1}{2i-1}\prod_{k=1}^{i}(2k-1)\frac{1}{i!}\frac{1}{2^i}\prod_{k=1}^{i}2k \\
&= \frac{1}{i!i!}\frac{1}{4^i}(-1)^{i-1}\frac{1}{2i-1}\prod_{k=1}^{i}(2k-1)(2k) \\
&= \frac{1}{4^i}(-1)^{i-1}\frac{1}{2i-1}\frac{1}{i!(2i-i)!}(2i)! \\
&= \frac{1}{4^i}(-1)^{i-1}\frac{1}{2i-1}\binom{2i}{i}.
\end{aligned}$$

Inserting these results into the definition of C_- yields

$$\begin{aligned}
C_-(x) &= \frac{1}{2x}\left(1-\sum_{i=0}^{\infty}\frac{1}{4^i}(-1)^{i-1}\frac{1}{2i-1}\binom{2i}{i}(-4x)^i\right) \\
&= \frac{1}{2x}\sum_{i=1}^{\infty}\frac{1}{2i-1}\binom{2i}{i}x^i \\
&= \sum_{i=0}^{\infty}\frac{1}{2}\frac{1}{2i+1}\binom{2i+2}{i+1}x^i \\
&= \sum_{i=0}^{\infty}\frac{1}{2}\frac{1}{2i+1}\frac{(2i+1)(2i+2)}{(i+1)(i+1)}\binom{2i}{i}x^i \\
&= \sum_{i=0}^{\infty}\frac{1}{i+1}\binom{2i}{i}x^i.
\end{aligned}$$

This finally gives the closed-form expression

$$C_i = \frac{1}{i+1}\binom{2i}{i} = \frac{(2i)!}{i!(i+1)!}.$$

Appendix C

Proof of the equivalence of matrix and hybrid formulation

Assume, that X, Y, Z satisfy the hybrid formulation (H1)-(H4).

We proof that (M1) holds by contradiction: Assume $\exists\ j \in \{2, \dots, N_C\},\ i \in \{1, \dots, j\}$ with

$$\sum_{m=1}^{j-i} X(i, j-m) + \sum_{m=1}^{i-1} X(i-m, j-m) < X(i,j).$$

This implies

$$\sum_{m=1}^{j-i} X(i, j-m) = 0 \tag{C.1}$$

$$\sum_{m=1}^{i-1} X(i-m, j-m) = 0 \tag{C.2}$$

$$X(i,j) = 1 \tag{C.3}$$

Inserting (C.3) into (H3) yields:

$$\begin{aligned}
&\exists\ l \in i, \dots, j-1: && Y(i,l,j) = 1 && \vee && \exists\ l \in j-i+1, \dots, j-1: && Z(l-(j-i), l, j) = 1 \\
\overset{\text{(H1)}}{\Rightarrow}\ &\exists\ l \in i, \dots, j-1: && X(i,l) = 1 && \vee && \exists\ l \in j-i+1, \dots, j-1: && X(l-(j-i), l) = 1 \\
\Rightarrow\ &\exists\ m \in 1, \dots, j-i: && X(i, j-m) = 1 && \vee && \exists\ m \in 1, \dots, i-1: && X(i-m, j-m) = 1
\end{aligned}$$

$\lightning$ (C.1),(C.2)

Further, (M2) holds: Let $j \in \{2, \dots, N_C\},\ i \in \{1, \dots, j\} : X(i,j) = 1$. Then by (H4):

$$\sum_{l=j+1}^{N_C} Y(i,j,l)(N_C + 1 - l) + \sum_{l=j+1}^{N_C} Z(i,j,l)(N_C + 1 - l) \geq N_C + 1 - j$$

Therefore

$$\exists\, l_d, l_b \in j+1,\dots,N_C:\ Y(i,j,l_d)=1, Z(i,j,l_b)=1:\ (N_C+1-l_d)+(N_C+1-l_b)\geq N_C+1-j$$
$$\overset{\text{(H2)}}{\Rightarrow} \exists\, l_d, l_b \in j+1,\dots,N_C:\ X(i,l_d)=1, X(i+(l_b-j),l_b)=1:\ (l_d-j)+(l_b-j)+j\leq N_C+1$$
$$\Rightarrow\ \exists\, m_d, m_b \in \{1,\dots,N_C-j\}:\ X(i,j+m_d)=1, X(i+m_b,j+m_b)=1:\ m_d+m_b+j\leq N_C+1.$$

Now, let X satisfy the matrix formulation (M1),(M2).
We define Y, Z as follows:

$$Y(i,j,l)=\begin{cases}1 & X(i,j)=1\wedge l=\min\limits_{\substack{l_d\in L\wedge\\ X(i,l_d)=1}} l_d\\ 0 & \text{else}\end{cases}, \qquad j=1,\dots,N_C-1,\ i=1,\dots,j$$

$$Z(i,j,l)=\begin{cases}1 & X(i,j)=1\wedge l=\min\limits_{\substack{l_b\in L\wedge\\ X(i+(l_b-j),l_b)=1}} l_b\\ 0 & \text{else}\end{cases}, \qquad j=1,\dots,N_C-1,\ i=1,\dots,j$$

with $L:=\{j+1,\dots,N_C\}$.
Then by construction (H1) and (H2) hold.
Further, (H3) holds: If $j\in\{2,\dots,N_C\}, i\in\{1,\dots,j\}: X(i,j)=1$ (else trivial), then by (M1):

$$\exists\, m_d\in\{1,\dots,j-i\}: X(i,j-m_d)=1\ \vee\ \exists\, m_b\in\{1,\dots,i-1\}: X(i-m_b,j-m_b)=1$$

We consider the first case, the second follows analogously. Then, by definition of Y, we must either have $Y(i,j-m_d,j)=1$ or $\exists\,\tilde{m}_d\in\{1,\dots,m_d-1\}: X(i,j-\tilde{m}_d)=1$. For the latter, we continue the argumentation analogously. Since m_d is bounded from below, we must reach the former at some point. Therefore, we obtain: $\exists\, l_d\in\{i,\dots,j-1\}: Y(i,l_d,j)=1$ or (from the second case) $\exists\, l_b\in\{j-i+1,\dots,j-1\}: Z(l_b-(j-i),l_b,j)=1$, which proves (H3).
Lastly, (H4) holds: Let $j\in\{2,\dots,N_C\},\ i\in\{1,\dots,j\}: X(i,j)=1$ (else trivial), then by definition of Y and Z:

$$\begin{aligned}
&\sum_{l=j+1}^{N_C} Y(i,j,l)(N_C+1-l)+\sum_{l=j+1}^{N_C} Z(i,j,l)(N_C+1-l)\\
&=(N_C+1-\min_{\substack{l_d\in L\wedge\\ X(i,l_d)=1}} l_d)+(N_C+1-\min_{\substack{l_b\in L\wedge\\ X(i+(l_b-j),l_b)=1}} l_b)\\
&=N_C+1-j+\left(N_C+1-\left[\min_{\substack{l_d\in L\wedge\\ X(i,l_d)=1}}(l_d-j)+\min_{\substack{l_b\in L\wedge\\ X(i+(l_b-j),l_b)=1}}(l_b-j)+j\right]\right)\\
&=N_C+1-j+\left(N_C+1-\left[\min_{\substack{m_d\in M\wedge\\ X(i,j+m_d)=1}} m_d+\min_{\substack{m_b\in M\wedge\\ X(i+m_b,m_b)=1}} m_b+j\right]\right)\\
&\overset{\text{(M2)}}{\geq} N_C+1-j
\end{aligned}$$

Appendix D

Generalization of the graph approach

By inserting (H1) and the fixed values $X(1,1) = 1,\ X(i,N_C) = 1,\ i = 1,\ldots,N_C$ into (H2), (H3) and (H4) we can derive a formulation that only involves the variables $Y(i,j,l), Z(i,j,l) \in \mathbb{B},\ (i,j,l) \in E$:

$$\sum_{l=2}^{N_C} Y(1,1,l) = 1$$
$$\sum_{l=2}^{N_C} Z(1,1,l) = 1$$

$\forall j = 2,\ldots,N_C - 1, i = 1,\ldots,j:$

$$\sum_{l=i}^{j-1} Y(i,l,j) \leq \sum_{l=j+1}^{N_C} Y(i,j,l)$$
$$\sum_{l=i}^{j-1} Y(i,l,j) \leq \sum_{l=j+1}^{N_C} Z(i,j,l)$$
$$\sum_{l=j-i+1}^{j-1} Z(l-(j-i),l,j) \leq \sum_{l=j+1}^{N_C} Y(i,j,l)$$
$$\sum_{l=j-i+1}^{j-1} Z(l-(j-i),l,j) \leq \sum_{l=j+1}^{N_C} Z(i,j,l)$$

$\forall i = 1,\ldots,j:$

$$\sum_{l=i}^{N_C-1} Y(i,l,N_C) \leq 1$$
$$\sum_{l=N_C-i+1}^{N_C-1} Z(l-(N_C-i),l,N_C) \leq 1$$

$\forall j = 2, \ldots, N_C - 1, i = 1, \ldots, j$:

$$\sum_{l=i}^{j-1} Y(i,l,j) + \sum_{l=j-i+1}^{j-1} Z(l-(j-i),l,j) \geq \sum_{l=j+1}^{N_C} Y(i,j,l)$$
$$\sum_{l=i}^{j-1} Y(i,l,j) + \sum_{l=j-i+1}^{j-1} Z(l-(j-i),l,j) \geq \sum_{l=j+1}^{N_C} Z(i,j,l)$$

$\forall i = 1, \ldots, N_C$:

$$\sum_{l=i}^{N_C-1} Y(i,l,N_C) + \sum_{l=N_C-i+1}^{N_C-1} Z(l-(N_C-i),l,N_C) \geq 1$$

$$\sum_{l=2}^{N_C} (Y(1,1,N_C) + Z(1,1,N_C))(N_C+1-l) \geq N_C$$

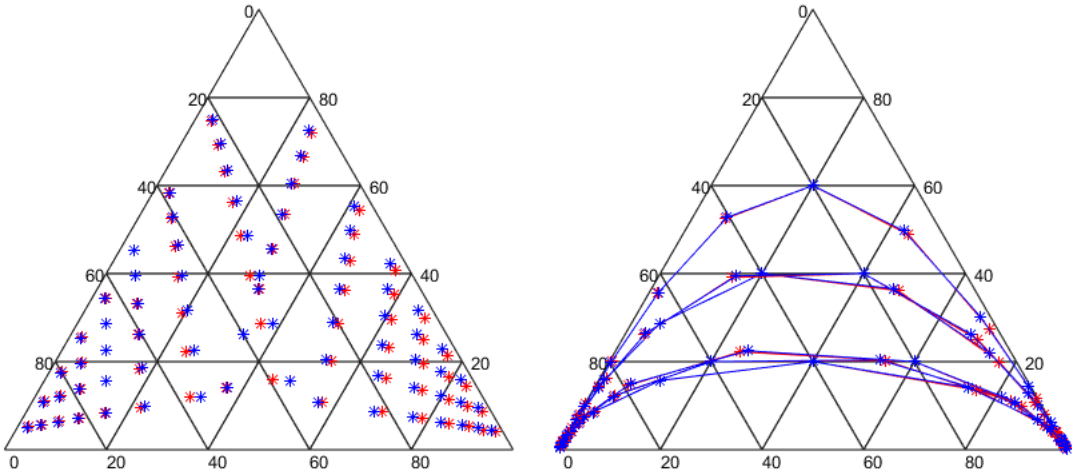
$\forall j = 2, \ldots, N_C - 1, i = 1, \ldots, j$:

$$\sum_{l=j+1}^{N_C} (Y(i,j,l) + Z(i,j,l)(N_C+1-l) \geq (N_C+1-j) \sum_{l=i}^{j-1} Y(i,l,j)$$
$$\sum_{l=j+1}^{N_C} (Y(i,j,l) + Z(i,j,l)(N_C+1-l) \geq (N_C+1-j) \sum_{l=j-i+1}^{j-1} Z(i+(l-j),l,j)$$

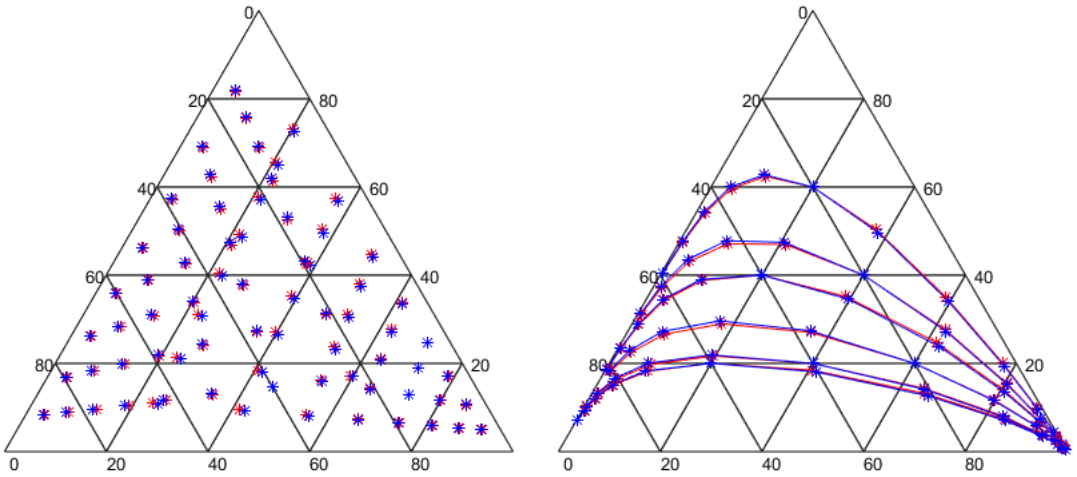
By utilizing the insights gained from the matrix approach we have derived a formulation that only involves the variables Y, Z and is completely general in N_C. This can be interpreted as an improvement of the original graph approach, established in Section 4.2.1.

Appendix E

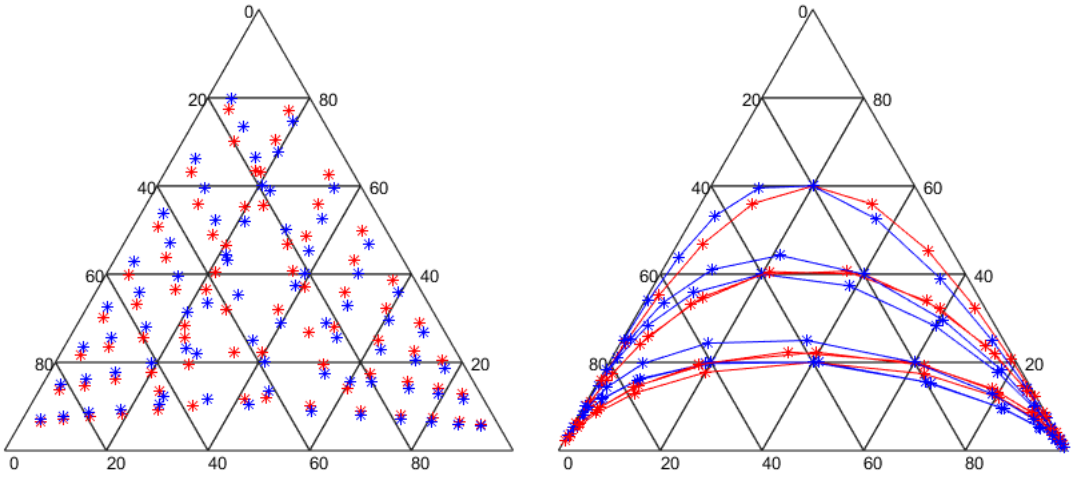
Additional numerical results



(a) A123



(b) A234



(c) C345

Figure E.1: Comparison of equilibrium points and curves resulting from the non-ideal (red) and constant relative volatility (blue) vapour liquid equilibrium models.

		CONOPT					filter					Knitro				
		A234	B235	B245	C235	C345	A234	B235	B245	C235	C345	A234	B235	B245	C235	C345
"rigorous"	β															
"normal"																
Split 1	0.95	12	17	15	16	11	31	196	34	46	15	36	58	20	62	12
	0.99	12	infeas	17	15	infeas	80	1001	58	465	it limit	10	infeas	9	18	infeas
Split 2	0.95	infeas	18	infeas	14	12	it limit	31	it limit	28	18	infeas	50	infeas	29	10
	0.99	infeas	11	infeas	10	infeas	it limit	607	it limit	78	1001	infeas	10	infeas	8	infeas
"large"																
Split 1	0.95	14	18	20	11	15	41	64	40	30	48	34	96	58	39	127
	0.99	9	12	9	9	9	26	infeas	30	28	24	8	11	8	9	9
Split 2	0.95	19	16	21	17	18	44	50	64	60	47	65	91	89	80	79
	0.99	infeas	9	14	11	11	infeas	40	it limit	26	161	infeas	12	13	10	10
"CMO"																
"normal"																
Split 1	0.95	23	22	12	18	28	9	infeas	infeas	infeas	11	106	infeas	10000	199	320
	0.99	11	infeas	12	15	11	8	infeas	infeas	infeas	12	97	infeas	314	infeas	10000
Split 2	0.95	infeas	18	infeas	17	25	infeas	infeas	infeas	9	20	infeas	129	infeas	infeas	173
	0.99	infeas	11	infeas	9	infeas	infeas	infeas	infeas	7	infeas	infeas	infeas	infeas	infeas	infeas
"large"																
Split 1	0.95	17	34	13	26	19	infeas	infeas	11	18	14	62	infeas	146	164	10000
	0.99	12	11	11	9	9	infeas	9	7	infeas	7	187	infeas	268	infeas	infeas
Split 2	0.95	26	13	22	14	15	19	15	infeas	12	23	infeas	98	infeas	infeas	240
	0.99	infeas	11	14	13	11	infeas	7	infeas	8	11	infeas	64	infeas	194	10000

Table E.1: Results for 16 test scenarios and remaining mixtures for the two single column optimization problems. Number of iterations (or termination message, in case of non-optimality) for different solvers on the NEOS Server. Selected solvers for further benchmarking (bold).

		direct	indirect	prefrac
"rigorous"	β			
"normal"				
	0.8	147.14	159.09	infeas
	0.9	**185.33**	192.08	infeas
	0.95	236.43	**231.55**	infeas
	0.99	647.96	**527.22**	infeas
	0.995	2094.5	**1480.94**	infeas
"large"				
	0.8	128.97	148.99	**99.95**
	0.9	151.78	166.45	**127.5**
	0.95	160.97	178.01	**154.49**
	0.99	**181.41**	192.45	386.49
	0.995	**196.41**	200.35	error

Table E.2: Results for 10 test scenarios for the single superstructure optimization problem (only "rigorous", "NLP") using fixed sequences and corresponding initial values and increasing purity parameter. Objective value (or termination message, in case of non-optimality) for CONOPT solver. Approximation of the globally optimal solution (bold).

		"NLP"						"MINLP"					
		CONOPT			filter			Bonmin			MINLP		
	normal	direct	indirect	prefrac	direct	indirect	prefrac	direct	indirect	prefrac	direct	indirect	prefrac
"rigorous"	β												
"normal"													
	0.8	143.32	159.09	infeas	143.32	159.09	141.63	147.14	159.09	infeas	147.14	159.09	141.62
		mix	indirect		mix	indirect	prefrac	direct	indirect		direct	indirect	prefrac
	0.9	185.33	192.08	infeas	185.33	192.08	234.16	185.33	192.08	723.48	185.33	192.08	723.48
		direct	indirect		direct	indirect	s. indirect	direct	indirect	prefrac	direct	indirect	prefrac
	0.95	236.43	231.55	236.43	236.43	231.55	400.74	236.43	231.55	infeas	236.43	231.55	infeas
		direct	indirect	direct	direct	indirect	s. indirect	direct	indirect		direct	indirect	
	0.99	647.96	527.22	infeas	647.96	527.22	527.22	infeas	527.22	infeas	647.96	527.22	infeas
		direct	indirect		direct	indirect	indirect		indirect		direct	indirect	
	0.995	2094.5	infeas	infeas	2094.5	1480.94	1480.94	infeas	1480.94	infeas	2094.5	1480.94	infeas
		direct			direct	indirect	indirect		indirect		direct	indirect	
"large"													
	0.8	116.37	99.14		116.37	148.99	99.14	128.97	140.71	infeas	128.97	140.71	99.95
		mix	$\sim$ prefrac	$\sim$ s. indirect	mix	indirect	$\sim$ prefrac	direct	indirect		direct	indirect	prefrac
	0.9	145.74	127.6	131.3	145.74	166.45	127.6	151.78	166.45	infeas	151.78	166.45	127.6
		mix	prefrac	infeas	mix	indirect	prefrac	direct	indirect		direct	indirect	prefrac
	0.95	160.41	178.08	178.08	160.41	178.08	154.49	160.97	178.08	infeas	160.97	178.01	154.49
		$\sim$ direct	indirect	indirect	$\sim$ direct	indirect	prefrac	direct	indirect		direct	indirect	prefrac
	0.99	181.41	192.45	infeas	181.41	192.45	235.59	181.41	192.45	infeas	181.41	192.45	386.24
		direct	indirect		direct	indirect	s. indirect	direct	indirect		direct	indirect	prefrac
	0.995	196.41	200.35	infeas	196.41	200.35	279.61	196.41	200.35	infeas	196.41	200.35	5525.9
		direct	indirect		direct	indirect	s. indirect	direct	indirect		direct	indirect	prefrac

Table E.3: Results for 10 test scenarios for the two superstructure optimization problems (only "rigorous","NLP"/"MINLP") using different initial values. Objective value (or termination message, in case of non-optimality) and sequence for different solvers on the NEOS Server.

		CONOPT		filter		MINLP	
			increasing β		increasing β		increasing β
"CMO"	β						
"normal"							
	0.8	infeas	infeas	157.76 prefrac	157.76 prefrac	155.61 direct	155.61 direct
	0.9	425.83 $\sim$ prefrac	infeas	276.65 s. indirect	infeas	196.15 direct	196.15 direct
	0.95	infeas	infeas	260.99 indirect	infeas	infeas	250.12 direct
	0.99	infeas	infeas	infeas	infeas	697.16 direct	error
	0.995	2395.5 indirect	infeas	infeas	infeas	infeas	infeas
"large"							
	0.8	106.83 $\sim$ prefrac	106.83 $\sim$ prefrac	infeas	infeas	133.07 direct	133.07 direct
	0.9	133.82 prefrac	133.82 prefrac	infeas	infeas	181.27 indirect	159 direct
	0.95	195.49 s. direct	161.99 prefrac	infeas	infeas	infeas	169.11 direct
	0.99	450.83 s. direct	430.62 prefrac	infeas	infeas	infeas	188.63 direct
	0.995	infeas	340.23 s. indirect	infeas	infeas	289.35 indirect	202.29 direct

Table E.4: Results for 10 test scenarios for the two superstructure optimization problem (only "CMO", "NLP"/"MINLP") using initial values from distributing the mass flow, without and with increasing purity parameter. Objective value (or termination message, in case of non-optimality) and sequence for different solvers on the NEOS Server.

		CONOPT			filter		
		direct	indirect	prefrac	direct	indirect	prefrac
"CMO"	β						
"normal"							
	0.8	155.61	174.28	infeas	infeas	174.28	infeas
	0.9	196.15	212.00	infeas	infeas	212.00	infeas
	0.95	250.12	260.99	infeas	infeas	260.99	infeas
	0.99	697.16	669.27	infeas	infeas	669.27	infeas
	0.995	2566.17	error	infeas	infeas	infeas	infeas
"large"							
	0.8	133.07	infeas	106.85	133.07	153.54	106.85
	0.9	159.00	infeas	133.83	error	181.19	infeas
	0.95	169.11	infeas	161.99	infeas	infeas	infeas
	0.99	188.63	infeas	430.62	infeas	infeas	infeas
	0.995	202.29	infeas	5700.83	infeas	infeas	infeas

Table E.5: Results for 10 test scenarios for the single superstructure optimization problem (only "CMO", "NLP") using fixed sequences and corresponding initial values and increasing purity parameter. Objective value (or termination message, in case of non-optimality) for different solvers on the NEOS Server.

		CONOPT			filter			MINLP		
		direct	indirect	prefrac	direct	indirect	prefrac	direct	indirect	prefrac
"CMO"	β									
"normal"										
	0.8	152.72 mix	174.28 indirect	152.72 mix	infeas	174.28 indirect	157.76 prefrac	155.61 direct	174.28 indirect	157.76
	0.9	196.15 direct	212 indirect	196.15 direct	infeas	212 indirect	infeas	196.15 direct	212 indirect	infeas
	0.95	250.12 direct	260.99	infeas	infeas	260.99 indirect	infeas	250.12 direct	260.99	infeas
	0.99	697.16 direct	infeas	697.16 direct	infeas	669.27 indirect	infeas	infeas	error	infeas
	0.995	2566.17 direct	infeas	2566.17 direct	infeas	infeas	infeas	infeas	infeas	infeas
"large"										
		direct	indirect	prefrac	direct	indirect	prefrac	direct	indirect	prefrac
	0.8	132.5 $\sim$ direct	152.93 $\sim$ indirect	106.83 $\sim$ prefrac	106.83 $\sim$ prefrac	infeas	106.83 $\sim$ prefrac	133.07 direct	infeas	106.85 prefrac
	0.9	infeas	181.19 indirect	181.83 $\sim$ s. indirect	infeas	infeas	infeas	159 direct	infeas	181.19 indirect
	0.95	168.49 $\sim$ direct	193.24 indirect	206.51 s. indirect	infeas	infeas	infeas	169.11 direct	infeas	161.99 prefrac
	0.99	188.63 direct	212.33 indirect	212.33 indirect	infeas	infeas	infeas	188.63 direct	infeas	infeas
	0.995	202.29 direct	223.77 indirect	340.23 s. indirect	infeas	infeas	infeas	error	infeas	infeas

Table E.6: Results for 10 test scenarios for the two superstructure optimization problems (only "CMO", "NLP"/"MINLP") using different initial values. Objective value (or termination message, in case of non-optimality) and sequence for different solvers on the NEOS Server.

x_1^F	x_2^F	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
0.1	direct	115.21	130.25	145.05	159.69	174.25	188.75	203.20	217.62
	indirect	105.84	124.36	142.18	159.59	176.75	193.73	210.58	227.34
	prefrac	***107.60***	***115.58***	***123.57***	***131.56***	***139.54***	***147.53***	***155.52***	***163.50***
0.2	direct	119.59	135.39	150.64	165.62	180.42	195.12	209.74	
	indirect	120.32	140.99	160.11	178.45	196.31	213.85	231.17	
	prefrac	***109.88***	***117.86***	***125.85***	***133.84***	***141.82***	***149.81***	***157.8***	
0.3	direct	124.96	141.27	156.83	172.02	187.00	201.84		
	indirect	137.59	159.71	179.70	198.66	217.01	234.96		
	prefrac	***112.16***	***120.15***	***128.13***	***136.12***	***144.1***	***152.09***		
0.4	direct	131.32	147.81	163.52	178.84	193.93			
	indirect	157.68	180.32	200.73	220.05	238.70			
	prefrac	***129.39***	***133.63***	***137.87***	***142.11***	***146.38***			
0.5	direct	**138.44**	**154.88**	170.62	185.99				
	indirect	179.94	202.42	222.93	242.39				
	prefrac	155.65	*159.89*	**164.13**	***168.37***				
0.6	direct	**146.05**	**162.33**	**178.04**					
	indirect	203.57	225.62	246.02					
	prefrac	*181.9*	*186.14*	*190.38*					
0.7	direct	**153.96**	**170.06**						
	indirect	228.06	249.58						
	prefrac	*208.16*	*212.4*						
0.8	direct	**162.05**							
	indirect	*253.06*							
	prefrac	234.41							

Table E.7: Results for mixture A123 and test scenarios arising from varying the feed composition for the matrix-based enumeration strategy with the `Knitro solver`, as in 8.11. Assumed globally optimal solution (bold). Solution obtained by the simultaneous optimization with the `Bonmin` solver (italic). If the solver stopped at an infeasible point, no solution is marked.

Bibliography

[1] J. L. Humphrey and A. F. Siebert, "Separation technologies; an opportunity for energy savings," *Chemical Engineering Progress*, vol. 88:3, 1992.

[2] Umweltbundesamt, "Branchenabhängiger Energieverbrauch des verarbeitenden Gewerbes," 08.04.2018. [Online]. Available: https://www.umweltbundesamt.de/daten/umwelt-wirtschaft/industrie/branchenabhaengiger-energieverbrauch-des

[3] J. Richardson, J. Harker, and J. R. Backhurst, *Coulson and Richardson's Chemical Engineering*, 5th ed. Elsevier, 2002.

[4] Wikimedia Commons, "Binary boiling point diagram," 22.05.2018. [Online]. Available: https://commons.wikimedia.org/wiki/File:Binary_Boiling_Point_Diagram.PNG

[5] ——, "Tray distillation tower," 04.06.2018. [Online]. Available: https://commons.wikimedia.org/wiki/File:Tray_Distillation_Tower_EN.svg

[6] S. M. Cheah, "Plant operations: Trays types," 08.04.2001. [Online]. Available: http://www.separationprocesses.com/Operations/POT_Chp02a.htm

[7] ——, "Tray vs. packing," 08.04.2001. [Online]. Available: http://www.separationprocesses.com/Operations/POT_Chp02h.htm

[8] H. Z. Kister, *Distillation design*. McGraw-Hill, 2013.

[9] C. J. King, *Separation processes*, 2nd ed. McGraw-Hill, 1980.

[10] Wikimedia Commons, "Mccabe-thiele diagram," 04.06.2018. [Online]. Available: https://commons.wikimedia.org/wiki/File:McCabe-Thiele_diagram.svg

[11] A. J. V. Underwood, "Fractional distillation of multi-component mixtures - calculation of minimum reflux ratio," *Journal of the Institute of Petroleum*, vol. 32, pp. 614–626, 1946.

[12] ——, "Fractional distillation of multicomponent mixtures," *Chemical Engineering Progress*, vol. 44, pp. 603–614, 1948.

[13] R. N. Shiras, D. N. Hanson, and C. H. Gibson, "Calculation of minimum reflux in distillation columns," *Industrial & Engineering Chemistry*, vol. 42, no. 5, pp. 871–876, 1950.

[14] I. J. Halvorsen and S. Skogestad, "Analytic expressions for minimum energy consumption in multicomponent distillation: A revisit of the underwood equations," in *AIChE Annual Meeting*, AIChE, Ed., 1999.

[15] ——, "Minimum energy consumption in multicomponent distillation. 1. vmin diagram for a two-product column," *Industrial & Engineering Chemistry Research*, vol. 42, no. 3, pp. 596–604, 2003.

[16] HyperTVT ETH Zürich, "Gilliland correlation," 07.10.2003. [Online]. Available: http://www.hyper-tvt.ethz.ch/gif/gilliland2.gif

[17] H. W. Kuhn and A. W. Tucker, "Nonlinear programming," in *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*. University of California Press, 1951, pp. 481–492.

[18] J. Gauvin, "A necessary and sufficient regularity condition to have bounded multipliers in nonconvex programming," *Mathematical Programming*, vol. 12, no. 1, pp. 136–138, 1977.

[19] G. Wachsmuth, "On licq and the uniqueness of lagrange multipliers," *Operations Research Letters*, vol. 41, no. 1, pp. 78–80, 2013.

[20] R. Courant, "Variational methods for the solution of problems of equilibrium and vibrations," *Bulletin of the American Mathematical Society*, vol. 49, no. 1, pp. 1–23, 1943.

[21] M. R. Hestenes, "Multiplier and gradient methods," *Journal of Optimization Theory and Applications*, vol. 4, no. 5, pp. 303–320, 1969.

[22] N. Karmarkar, "A new polynomial-time algorithm for linear programming," *Combinatorica*, vol. 4, pp. 373–395. [Online]. Available: https://link.springer.com/article/10.1007/BF02579150

[23] S. Mehrotra, "On the implementation of a primal-dual interior point method," *SIAM Journal on Optimization*, vol. 2, no. 4, pp. 575–601, 1992.

[24] U. M. G. Palomares and O. L. Mangasarian, "Superlinearly convergent quasi-newton algorithms for nonlinearly constrained optimization problems," *Mathematical Programming*, vol. 11, no. 1, pp. 1–13, 1976.

[25] J. Nocedal and S. J. Wright, *Numerical Optimization*, 2nd ed. Springer Science+Business Media LLC, 2006.

[26] R. J. Dakin, "A tree-search algorithm for mixed integer programming problems," *The Computer Journal*, vol. 8, no. 3, pp. 250–255, 1965.

[27] R. A. Stubbs and S. Mehrotra, "A branch-and-cut method for 0-1 mixed convex programming," *Mathematical Programming*, vol. 86, no. 3, pp. 515–532, 1999.

[28] M. A. Duran and I. E. Grossmann, "An outer-approximation algorithm for a class of mixed-integer nonlinear programs," *Mathematical Programming*, vol. 36, no. 3, pp. 307–339, 1986.

[29] A. M. Geoffrion, "Generalized benders decomposition," *Journal of Optimization Theory and Applications*, vol. 10, no. 4, pp. 237–260, 1972.

[30] T. Westerlund and F. Pettersson, "An extended cutting plane method for solving convex minlp problems," *Computers & Chemical Engineering*, vol. 19, pp. 131–136, 1995.

[31] J. Lee and S. Leyffer, Eds., *Mixed Integer Nonlinear Programming*. Springer New York, 2012.

[32] S. Scholtes and M. Stöhr, "How stringent is the linear independence assumption for mathematical programs with complementarity constraints?" *Mathematics of Operations Research*, vol. 26, no. 4, pp. 851–863, 2001.

[33] M. Anitescu, P. Tseng, and S. J. Wright, "Elastic-mode algorithms for mathematical programs with equilibrium constraints: Global convergence and stationarity properties," *Mathematical Programming*, vol. 110, no. 2, pp. 337–371, 2007.

[34] R. Fletcher and S. Leyffer, "Numerical experience with solving mpecs as nlps," *University of Dundee Report NA*, vol. 2002, no. 210.

[35] B. T. Baumrucker, J. G. Renfro, and L. T. Biegler, "Mpec problem formulations and solution strategies with chemical engineering applications," *Computers & Chemical Engineering*, vol. 32, no. 12, pp. 2903–2913, 2008.

[36] G. L. Nemhauser and L. A. Wolsey, *Integer and combinatorial optimization*. Wiley, 1999.

[37] S. Lee and I. E. Grossmann, "New algorithms for nonlinear generalized disjunctive programming," *Computers & Chemical Engineering*, vol. 24, no. 9-10, pp. 2125–2141, 2000.

[38] M. Türkay and I. E. Grossmann, "Logic-based minlp algorithms for the optimal synthesis of process networks," *Computers & Chemical Engineering*, vol. 20, no. 8, pp. 959–978, 1996.

[39] I. E. Grossmann and J. P. Ruiz, "Generalized disjunctive programming: A framework for formulation and alternative algorithms for minlp optimization," in *Mixed Integer Nonlinear Programming*, J. Lee and S. Leyffer, Eds. Springer New York, 2012, vol. 154, pp. 93–115.

[40] R. W. Thompson and C. J. King, "Systematic synthesis of separation schemes," *AIChE Journal*, vol. 18, no. 5, pp. 941–948, 1972.

[41] A. Giridhar and R. Agrawal, "Synthesis of distillation configurations: I. characteristics of a good search space," *Computers & Chemical Engineering*, vol. 34, no. 1, pp. 73–83, 2010.

[42] R. Agrawal, "Synthesis of multicomponent distillation column configurations," *AIChE Journal*, vol. 49, no. 2, pp. 379–401, 2003.

[43] A. Giridhar and R. Agrawal, "Synthesis of distillation configurations. ii: A search formulation for basic configurations," *Computers & Chemical Engineering*, vol. 34, no. 1, pp. 84–95, 2010.

[44] V. H. Shah and R. Agrawal, "A matrix method for multicomponent distillation sequences," *AIChE Journal*, vol. 56, no. 7, pp. 1759–1775, 2010.

[45] R. Sargent and K. Gaminibandara, "Optimum design of plate distillation columns," in *Optimization in action*, L. C. Dixon, Ed. Academic Press, 1976.

[46] U. Nallasivam, V. H. Shah, A. A. Shenvi, M. Tawarmalani, and R. Agrawal, "Global optimization of multicomponent distillation configurations: 1. need for a reliable global optimization algorithm," *AIChE Journal*, vol. 59, no. 3, pp. 971–981, 2013.

[47] H. Yeomans and I. E. Grossmann, "A systematic modeling framework of superstructure optimization in process synthesis," *Computers & Chemical Engineering*, vol. 23, no. 6, pp. 709–731, 1999.

[48] J. A. Caballero and I. E. Grossmann, "Design of distillation sequences: From conventional to fully thermally coupled distillation systems," *Computers & Chemical Engineering*, vol. 28, no. 11, pp. 2307–2329, 2004.

[49] L. Zhang and A. A. Linninger, "Towards computer-aided separation synthesis," *AIChE Journal*, vol. 52, no. 4, pp. 1392–1409, 2006.

[50] ——, "Temperature collocation algorithm for fast and robust distillation design," *Industrial & Engineering Chemistry Research*, vol. 43, no. 12, pp. 3163–3182, 2004.

[51] University of Wisconsin - Madison, "Neos server faq." [Online]. Available: https://neos-guide.org/content/FAQ

[52] W. Gropp and J. Moré, "Optimization environments and the neos server," in *Approximation theory and optimization*, M. D. Buhmann, Ed. Cambridge University Press, 1997, pp. 167–182.

[53] J. Czyzyk, M. P. Mesnier, and J. J. More, "The neos server," *IEEE Computational Science and Engineering*, vol. 5, no. 3, pp. 68–75, 1998.

[54] E. Dolan, "The neos server 4.0 administrative guide," Mathematics and Computer Science Division, Argonne National Laboratory, Tech. Rep. ANL/MCS-TM-250.

[55] University of Wisconsin - Madison, "Neos solvers," 23.04.2018. [Online]. Available: https://neos-server.org/neos/solvers/index.html

[56] R. Fourer, "Algebraic modeling languages for optimization," in *Encyclopedia of operations research and management science*, S. I. Gass and M. Fu, Eds. Springer US, 2013.

[57] M. B. Villarino, "On the convergence of the catalan generating function," 2015. [Online]. Available: http://arxiv.org/pdf/1511.08555