Faculteit Industriële Ingenieurswetenschappen

master in de industriële wetenschappen: chemie

Masterthesis

Automated compartmental modelling of chemical reactors

Rutger Muermans

Scriptie ingediend tot het behalen van de graad van master in de industriële wetenschappen: chemie

PROMOTOR :
Prof. dr. ir. Mumin Enis LEBLEBICI

BEGELEIDER :
ing. Ulderico DI CAPRIO

Gezamenlijke opleiding UHasselt en KU Leuven

Universiteit Hasselt | Campus Diepenbeek | Faculteit Industriële Ingenieurswetenschappen | Agoralaan Gebouw H - Gebouw B | BE 3590 Diepenbeek

Universiteit Hasselt | Campus Diepenbeek | Agoralaan Gebouw D | BE 3590 Diepenbeek
Universiteit Hasselt | Campus Hasselt | Martelarenlaan 42 | BE 3500 Hasselt

2024
2025

Faculteit Industriële Ingenieurswetenschappen

master in de industriële wetenschappen: chemie

Masterthesis

Automated compartmental modelling of chemical reactors

Rutger Muermans

Scriptie ingediend tot het behalen van de graad van master in de industriële wetenschappen: chemie

PROMOTOR :

Prof. dr. ir. Mumin Enis LEBLEBICI

BEGELEIDER :

ing. Ulderico DI CAPRIO

UHASSELT KU LEUVEN

Preface

This thesis marks the final step of my master's in chemical engineering at UHasselt. The research presented here focuses on the modelling of optimal adiabatic reactor configurations for exothermic reactions using evolutionary algorithms. Writing this thesis has been both challenging and educational, allowing me to further develop my interest in process optimization and computational modelling. The code developed during this thesis can be accessed via the following GitLab repository: https://gitlab.kuleuven.be/cipt/master-theses/rutger_muermans_2025

I would like to extend my gratitude to everyone who accompanied and supported me along the way. I would like to begin by thanking my promoter, Prof. Leblebici, and my supervisor, Dr. Di Caprio, for their helpful support and advice.

I also wish to thank my girlfriend and family, whose unwavering support, interest, and encouragement have meant a great deal to me.

Looking back, I am proud of the results achieved and grateful for the learning process it entailed. I hope this thesis will contribute to further developments in the field of chemical reactor design.

Enjoy reading!

Rutger Muermans
Geleen, June 2025

Table of contents

List of Tables

List of Figures

Abstract

Designing optimal reactor configurations for exothermic reactions is complex and requires balancing multiple, often conflicting objectives. Traditional methods can be time-consuming and may not efficiently explore the wide space of possible configurations, especially when criteria like reactor volume, intercooling needs, and process complexity must be considered.

This thesis develops a two-stage genetic algorithm for optimizing adiabatic reactor configurations. The goal is to maximize the efficiency of the reactor configuration, while minimizing total reactor volume, intercooling demand, and the number of adiabatic steps, ensuring a total conversion of 0.95.

First, a multi-objective NSGA-II algorithm is used to determine the optimal sequence of adiabatic steps. In the second stage, a single-objective algorithm selects the most suitable reactor type or combination for each step. The method was applied to four case studies with varying weightings of the objectives. Results show that the algorithm adapts its strategy to shifting priorities: emphasizing reactor volume reduction leads to more adiabatic steps, prioritizing intercooling favors higher inlet temperatures, and minimizing complexity results in fewer steps but larger reactors. The algorithm efficiently generates balanced configurations in about 10 minutes per weight combination, which is much faster than manual methods that can take a full day and may still miss the optimum, making it a valuable tool for automated modelling and decision-making in chemical engineering.

Dutch Abstract

Het ontwerpen van optimale reactorconfiguraties voor exotherme reacties is complex en vereist het balanceren van vaak conflicterende doelstellingen. Traditionele methoden zijn tijdrovend en verkennen de grote ruimte aan configuraties vaak niet efficiënt, zeker wanneer criteria zoals reactorvolume, tussenkoeling en procescomplexiteit samen overwogen moeten worden.

Deze thesis ontwikkelt een tweetraps genetisch algoritme voor het optimaliseren van adiabatische reactorconfiguraties. Het doel is de efficiëntie van de configuratie te maximaliseren, terwijl het totale reactorvolume, de tussenkoeling en het aantal adiabatische stappen geminimaliseerd worden, met als einddoel een totale conversie van 0,95.

Allereerst bepaalt een multi-objectief NSGA-II-algoritme de optimale volgorde van adiabatische stappen. In de tweede fase selecteert een single-objectief algoritme het meest geschikte reactortype per stap. De methode werd getest op vier casestudies met verschillende doelstellingsgewichten. Resultaten tonen dat het algoritme zich aanpast aan prioriteiten: focus op volumevermindering leidt tot meer stappen, minder tussenkoeling bevordert hogere starttemperaturen, en minder adiabatische stappen resulteren in grotere reactors. Het algoritme levert efficiënte configuraties op in circa 10 minuten per gewichtscombinatie wat veel sneller is dan handmatige methoden, die een dag kunnen duren en vaak suboptimaal blijven. Dit maakt het project tot een handige tool voor geautomatiseerd modelleren en besluitvorming in de chemische procestechniek.

1. Introduction

Chemical reactors are central to numerous industrial processes, driving the conversion of raw materials into valuable products such as fuels, polymers, pharmaceuticals, and chemical intermediates. The performance and efficiency of these processes are strongly dependent on the reactor design and operating conditions. Within chemical process technology, the plug flow reactor (PFR) and the continuously stirred tank reactor (CSTR) are two commonly used ideal reactor types. However, their design is a complex task that often depends on simplifying assumptions and practical, experience-based intuitions.

Industrial processes often use non-isothermal reactors, in which temperature variations influence the reaction kinetics and product distribution [1, 2]. Depending on the nature of the chemical reaction, it can be endothermic or exothermic. In an endothermic process, heat from the environment is absorbed by the reaction mixture, while in an exothermic process energy is released [1, 2]. The way in which heat is managed within the process largely determines the temperature gradients and subsequently the conversion and selectivity of the reaction.

Reactors can be designed as adiabatic or non-adiabatic systems. In an adiabatic system, no heat is exchanged with the surroundings, meaning that the temperature of the reaction mixture changes solely due to the heat generated or consumed by the reaction [1, 2]. In an exothermic adiabatic system, the temperature of the reaction mixture rises as the conversion increases, while in an endothermic adiabatic system, the temperature gradually decreases [1, 2]. However, in a non-adiabatic reactor an active heat exchange is applied, for example by means of a heat exchanger [1, 2].

Traditional methods for designing reactor configurations are time-consuming and often suboptimal, as they rely on iterative calculations and the experience of the designer. This causes a problem as today's industry increasingly demands automated and optimized processes to minimize both costs and energy consumption. It is therefore necessary to develop a more efficient method that speeds up the design process and yields better configurations.

In recent years, computational optimization methods have become increasingly popular for solving complex engineering problems involving multiple objectives and nonlinear relationships. Among them, genetic algorithms (GAs) stand out as powerful and flexible tools, especially well-suited for exploring large and multi-dimensional design spaces [3, 4]. Inspired by natural selection, GAs evolve a population of candidate solutions over successive generations through selection, crossover, and mutation [3, 4, 5].

Each individual in the population represents a potential solution, which in this case is a specific reactor configuration [3, 4]. These individuals are evaluated using a fitness function that reflects key objectives, such as maximizing conversion, minimizing reactor volume, limiting intercooling, and reducing the number of reactors, translating the problem into a multi-objective one [3, 4, 5]. The best-performing individuals are selected to generate new offspring, gradually improving the population [4].

To effectively manage the multi-objective nature of the problem, this thesis uses NSGA-II (Non-dominated Sorting Genetic Algorithm II), an advanced GA designed for multi-objective optimization [6, 7]. NSGA-II sorts individuals based on Pareto dominance and uses a crowding distance metric to maintain diversity across the solution space [6, 7]. This allows the algorithm to generate a diverse Pareto front of optimal reactor configurations that balance trade-offs between competing objectives [6, 7].

In the context of reactor design, GAs provide an effective way to explore a wide range of possible reactor configurations. These may vary in the number of reactors, the amount of heat exchange between the reactors, their individual starting and final temperatures, and the type of reactor used (e.g., PFR, CSTR, Hybrid or PFR with recycle). Each of these factors influences conversion, energy usage, and overall efficiency.

This thesis focuses on the development of a computational methodology that applies a genetic algorithm to automate the design of adiabatic reactor configurations for exothermic reactions. The central goal is to minimize the surface area under the Levenspiel plot. This is a graphical representation of the inverse reaction rate versus conversion which directly correlates with the required reactor volume [1, 8]. Additionally, the algorithm considers multiple objectives such as minimizing intercooling energy and the number of adiabatic steps, while ensuring that the final conversion reaches a desired target. This goal leads to the research question of this thesis: “How can a genetic algorithm be developed to determine an optimal adiabatic reactor configuration for a specific exothermic chemical reaction?”.

This report presents the implementation of a two-step algorithm, using standard genetic algorithms, to obtain the optimal reactor configuration discussed. It begins in Chapter 2 with the theoretical background, where the fundamental concepts related to chemical reaction kinetics, ideal reactor models (PFRs and CSTRs) and the Levenspiel plot are introduced. This chapter also explains the working principles of genetic algorithms, with a focus on the Non-dominated Sorting Genetic Algorithm II (NSGA-II), which is used in this research for multi-objective optimization.

Building on this foundation, Chapter 3 outlines the methodology used to implement the optimization approach. It describes the calculation of reaction kinetics, the implementation of the genetic algorithms, and the evaluation method used to assess the performance of the designed two-step algorithm.

Chapter 4 provides the simulation results and evaluates the performance of the developed algorithm. The optimized reactor configurations are analyzed and compared based on various objective weight combinations of surface area, intercooling requirements, and the number of adiabatic steps. Particular attention is given to how these different weight combinations affect the resulting configurations, with a focus on conversion, required total reactor volume, intercooling needs, and the number of reactor units. The goal is to assess whether the algorithm provides a logically optimal adiabatic configuration for each combination of weights.

Finally, Chapter 5 concludes the work. It also discusses the constraints of the current model and suggests possible directions for future research.

2. Literature study

This chapter provides the theoretical background required to understand and implement the optimization approach used in this thesis. First, the kinetic data of chemical reactions is presented, which serves as essential input for evaluating reactor performance and guiding the optimization process (Section 2.1). Next, different ideal reactor types are introduced, with a focus on how these impact conversion and reaction rate (Section 2.2). The concept of the Levenspiel plot is then explained as a graphical tool to evaluate reactor volume requirements based on reaction kinetics (Section 2.3). Following this, the operation of the genetic algorithm used to optimize reactor configurations is described in detail (Section 2.4). To assess trade-offs between competing objectives, the Pareto diagram is introduced (Section 2.5). Finally, the specific algorithm applied in this research, NSGA-II, is explained, highlighting how it enables multi-objective optimization (Section 2.6).

2.1 Kinetic data of chemical reactions

Chemical reactions can be categorized as either reversible or irreversible, depending on whether the products can be converted back into the reactants. In reversible reactions, both the forward and reverse reactions can occur [1, 9]. Under limiting conditions, these reactions reach an equilibrium, where the rates of the forward and reverse reactions are equal, and the concentrations of the reactants and products remain constant [1, 9]. The Water-Gas Shift Reaction (WGSR) serves as a typical example of a reversible reaction, as shown below [10].

$$CO + H_2O \rightleftharpoons CO_2 + H_2$$

In the WGSR, carbon monoxide and water react to produce carbon dioxide and hydrogen, but the reverse reaction also takes place under suitable conditions. In contrast, irreversible reactions proceed predominantly in one direction, converting reactants fully into products without a backward reaction [9]. A well-known example is the combustion of methane as is shown below [11].

$$CH_4 + 2\ O_2 \rightarrow CO_2 + 2\ H_2O$$

Here, the products are so stable that the reverse reaction is practically impossible. In any chemical reaction, the extent to which products are formed is limited by the reactant that is present in the lowest stoichiometric proportion relative to the balanced reaction equation [1, 12]. This substance is known as the limiting reactant. When this reactant is fully consumed, the reaction cannot continue, even if other reactants remain in excess.

Reactions also differ in their thermal characteristics. A reaction is called exothermic if it releases heat into the surroundings (i.e., the reaction enthalpy ΔH_r is negative) [1, 2]. Conversely, in an endothermic reaction, heat is absorbed (i.e., $\Delta H_r > 0$) [1, 2]. These thermal properties affect how temperature influences the equilibrium position of a reversible reaction.

Le Chatelier's Principle states that if the temperature of an equilibrium system is altered, the system will adjust itself to oppose the change [13]. For example, in an exothermic reaction, increasing the temperature shifts the equilibrium toward the reactants, reducing conversion. In contrast, for an endothermic reaction, higher temperatures favor the forward reaction, increasing product formation.

These effects can be quantified using thermodynamic principles. The equilibrium constant K at 298 Kelvin (K_{298}) can be derived from the standard Gibbs free energy at this temperature (ΔG^0_{298}) as is shown in Equation 1 [1, 2].

$$K_{298} = e^{\frac{-\Delta G°_{298}}{R \cdot T}} \quad (1)$$

Here, R is the gas constant and T represents the temperature in Kelvin. To evaluate how K changes with temperature, the van 't Hoff equation is used, shown in Equation 2 [1, 2].

$$K = K_{298} \cdot e^{\frac{-\Delta H_r}{R} \cdot (\frac{1}{T} - \frac{1}{298})} \quad (2)$$

Once the equilibrium constant at every temperature is known, the equilibrium conversion $X_{A,e}$ of the limiting reactant A can be calculated using Equation 3 [1, 2].

$$X_{Ae} = \frac{K}{K+1} \quad (3)$$

This expression is derived under the assumption of an elementary, reversible reaction of the type A⇌B [1].

These considerations occur under thermodynamic conditions, which assume that the system has sufficient time to reach equilibrium, effectively on an infinite time scale. However, in practical reactor design, kinetic factors often play a more decisive role. While thermodynamics determines the final equilibrium state of a reaction, indicating whether a reaction is favorable, kinetics describes the rate at which the reaction proceeds. Even a thermodynamically favorable reaction may proceed very slowly if the kinetics are limiting. Thus, a reaction may theoretically reach equilibrium, but in practice, this may only happen over impractically long timescales if the reaction rate is too low [1, 2].

The equilibrium line, as illustrated in Figure 1, represents the maximum achievable conversion at a given temperature. At any point along the equilibrium line, the forward and reverse reaction rates are equal, indicating that the system has reached dynamic equilibrium and no further increase in conversion is possible under those conditions [1, 13].

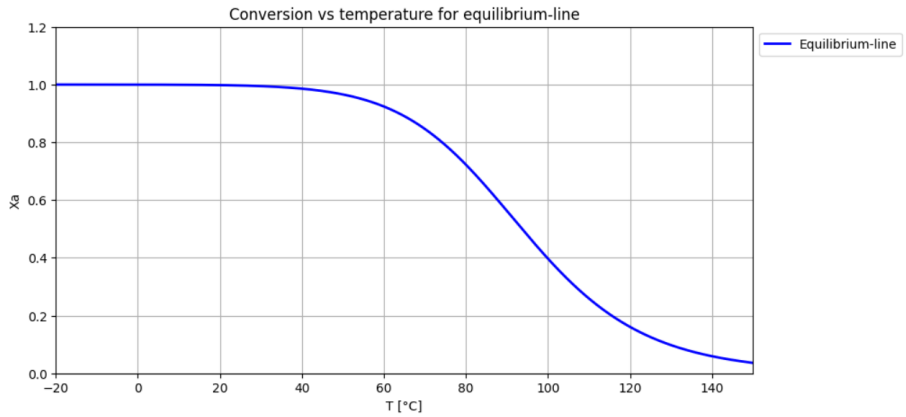


Figure 1: Equilibrium line for a reversible exothermic reaction

While thermodynamics shows the maximum conversion that can be reached at a given equilibrium, kinetics provides insight into how fast this conversion occurs. The rate at which equilibrium is achieved, and the pathway the reaction takes to reach this equilibrium, are determined by the kinetic properties of the reaction. The following section discusses the kinetics of chemical reactions, focusing on the factors that influence the rate of the reaction.

The kinetics of chemical reactions studies the rate at which reactions occur and the factors that influence this rate. A reaction can be described using an equation, where reactant and product concentrations vary over time. The rate is affected by several factors, such as pressure, temperature, reactant concentration, and the application of catalysts [1, 14]. The rate of a reaction is often described by a rate equation, which depends on the reaction pathway and mechanisms. A simple reaction can take the form A + B → C, where the reaction rate can be formulated as $-r_A = k \cdot [A]^m \cdot [B]^n$ [1, 14]. In this formula, k denotes the rate constant, while m and n represent the reaction orders. The rate constant k varies with temperature and is described by the Arrhenius equation, as shown in Equation 4 [1, 2, 15].

$$k = A \cdot e^{-\frac{E_a}{R \cdot T}} \tag{4}$$

In this equation, A represents the frequency factor, E_a denotes the activation energy, R is the gas constant, and T corresponds to the temperature in Kelvin [1, 2, 15]. When the frequency factor and activation energy for a given reaction are known, the reaction rate ($-r_A$) can be expressed as a function of temperature and conversion using Equation 5 [1].

$$-r_A = k_1 \cdot C_{A,0} \cdot (1 - X_A) - k_2 \cdot C_{A,0} \cdot X_A \tag{5}$$

Here, k_1 and k_2 present the rate constants for the forward and reverse reactions, respectively, $C_{A,0}$ represents the initial concentration of the limiting component, and X_A represents the conversion. An example of the equilibrium line with reaction rate lines for a reversible exothermic reaction is shown in Figure 2.

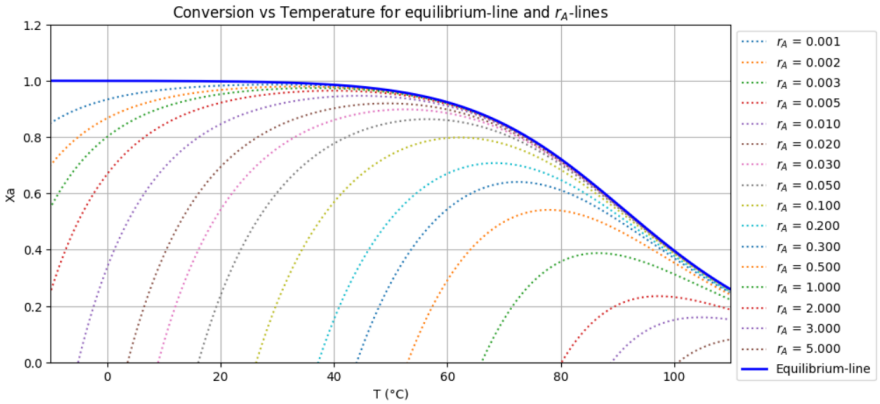


Figure 2: Equilibrium line with reaction rates for a reversible exothermic reaction

2.2 Reactor design and configuration

Chemical reactors are typically categorized into two idealized types: the plug flow reactor (PFR) and the stirred tank reactor (CSTR). These models approximate real reactor behavior, helping to predict conversion, selectivity, and energy efficiency [1, 16, 17]. The selection of a reactor is influenced by factors like reaction kinetics, process conditions, and heat and mass transfer requirements.

In a PFR, the flow is segregated, meaning that reactants move through the reactor as "plugs" with minimal mixing, leading to concentration gradients [16, 17]. In contrast, a CSTR features continuous mixing, ensuring uniform concentration and temperature throughout the reactor [17]. This difference in mixing behavior affects the reaction rate and reactor performance. When reactors are placed in series, such as a series of PFRs or a combination of PFRs and CSTRs, the performance and temperature control can be further optimized to achieve higher conversion and efficiency [1].

2.2.1 Plug flow reactor and continuous stirred tank reactor

A PFR is a tubular reactor where reactants flow continuously through the tube without mixing along the direction of flow [16]. This means that the concentration and temperature of the reactants change differentially along the reactor length, as each segment of the flow experiences a different amount of time for the reaction to occur [16].

To mathematically model this, the reactor is approximated as a series of infinitesimally small sections, where each section behaves as a small volume where a differential reaction takes place [16, 17]. This approximation allows the conversion to increase stepwise along the reactor's length, providing a simple but effective method for analyzing the reaction kinetics and performance within the PFR [1, 16, 17]. Figure 3 shows a schematic representation of the PFR.

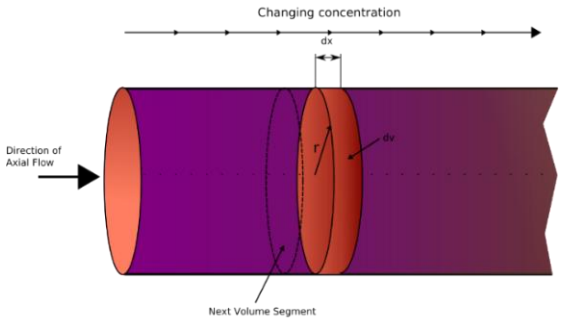


Figure 3: Schematic representation of a PFR [18]

A CSTR is an idealized reactor where the contents are continuously stirred, ensuring that the concentration and temperature are uniform throughout the reactor [1, 17]. In a CSTR, the reactants are continuously mixed with the reaction mixture, which results in a uniform concentration of reactants throughout the reactor [1, 17]. The outlet concentration of the reactants is assumed to be the same as the concentration within the reactor, which simplifies the analysis of the reaction [1, 17]. CSTRs are particularly useful for liquid-phase reactions or processes where maintaining a constant reactant concentration is beneficial [19]. However, because the reactants are well-mixed, their concentration within a CSTR is uniform and corresponds to the outlet concentration, which is relatively low compared to the higher average concentration in a PFR operating at the same conversion. The lower concentration in a CSTR results in a slower reaction rate because reaction rates are often dependent on the concentration of the reactants [1, 17]. Figure 4 shows a schematic representation of the CSTR.

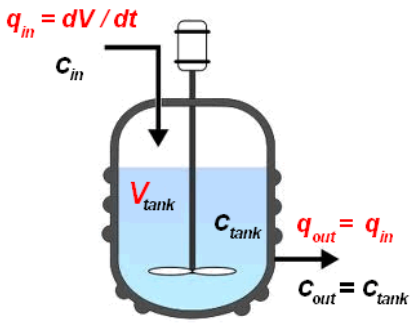


Figure 4: Schematic representation of a CSTR [20]

A variation of the PFR is the PFR with recycle, where a portion of the effluent reaction products is returned to the reactor inlet, as shown in Figure 5. The recycle stream reduces the segregation that is characteristic of a standard PFR [21]. By reintroducing part of the effluent back into the reactor, the recycle stream promotes more uniform mixing and can decrease the concentration gradient, making the reactor behave more similarly to a CSTR [21]. The extent to which this behavior resembles that of a CSTR depends on the recycle value, which is defined as the ratio between the amount of flow that goes into the recycle stream and the amount that continues as product [21]. Higher recycle values lead to more uniform conditions and a closer approximation to a CSTR.

In practical terms, "recycle" refers to the process of returning a portion of the reactor effluent, either the unreacted reactants or products, back into the reactor inlet [21]. A key advantage of a PFR with recycle is that it can result in a smaller required reactor volume compared to a traditional PFR [1]. This occurs because the recycle stream helps to homogenize the concentration and temperature profiles within the reactor, effectively reducing the concentration gradient along the reactor's length [1].

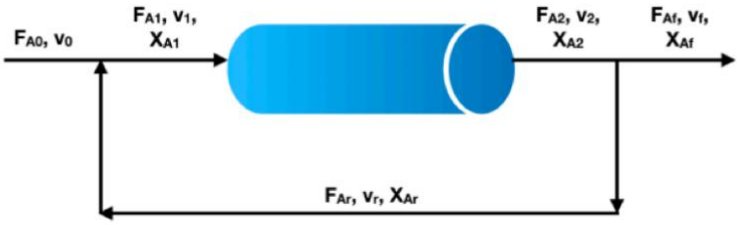


Figure 5: PFR with recycle [21]

2.2.2 Adiabatic and non-adiabatic reactors

Chemical reactors can be classified as either adiabatic or non-adiabatic. Adiabatic reactors do not transfer or remove heat to their surroundings [1, 2]. In exothermic reactions occurring in an adiabatic reactor, the reactor temperature increases as the conversion increases, while in endothermic reactions, the temperature decreases as the reaction progresses [1, 2]. Non-adiabatic reactors are cooled or heated to maintain the temperature within desired range [1, 2]. This can be useful to prevent excessive temperatures in exothermic reactions or to keep an endothermic reaction going.

In exothermic reactions occurring in an adiabatic reactor, the temperature within a reactor can increase significantly, which can lead to unwanted side reactions or a shift in the chemical equilibrium [1, 13]. One strategy to control this is to use intercooling between successive reactors in the series. In a series of adiabatic reactors, a heat exchanger can be placed after each reactor to reduce the temperature before the flow enters the next reactor. This helps to maintain the reaction within a favorable temperature range and achieve a higher conversion [1].

2.2.3 Adiabatic line in X_A vs. T diagram

This thesis focusses on the reactor modelling of exothermic adiabatic systems. When designing an adiabatic reactor configuration for exothermic reactions, understanding the correlation between conversion and temperature is important. As discussed in Paragraph 2.2, for adiabatic reactors, the temperature evolution as a function of conversion is influenced by the heat produced or consumed by the reaction. The adiabatic line, which represents the temperature change as a function of conversion in an adiabatic reactor, can be approximated as a straight line under certain assumptions [1]. These assumptions include a first-order reaction, constant specific heat capacity (c_p), and constant heat of reaction independent of temperature and conversion [1]. This line can be determined using Equation 6 [1].

$$X_A(T) = X_{A,start} + \frac{C_p}{\Delta H_r} \cdot (T - T_0) \tag{6}$$

As discussed in Paragraph 2.1, the equilibrium line represents the temperature at which the reaction reaches equilibrium for a given conversion. Beyond this line, the reaction cannot proceed further without external measures such as pressure increase or heat removal. To achieve maximum conversion, it is necessary to ensure that the temperature in the reactor does not exceed the equilibrium temperature, as this would limit further reaction progress.

One strategy to overcome this limitation is intercooling between reactors [1]. In multi-reactor configurations, intercooling between adiabatic reactors reduces the temperature of the reaction mixture, thereby limiting temperature build-up and allowing for increased conversion in the following reactor [1]. Figure 6 shows a block flow diagram of an adiabatic reactor configuration with two reactors and intercooling for an exothermic reaction.

This strategy of intercooling works well in exothermic reactions, where the temperature increase is a challenge. In contrast, for endothermic reactions, the temperature decreases as conversion increases, and intercooling is not required to the same extent. Instead, the reactor design may focus on maintaining a temperature that supports the forward reaction, typically requiring heat input to sustain the reaction rate [1].

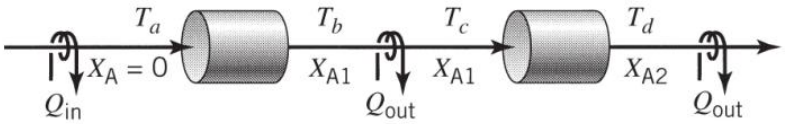


Figure 6: Block flow diagram of reactors in series with intercooling. Adapted from [1, p. 433].

Figure 7 shows an example of an adiabatic configuration with four adiabatic steps and intercoolers for an exothermic reaction. The final conversion X_A of this configuration is 0.85 at a temperature of 67°C. The adiabatic lines, with a slope of $\frac{C_p}{\Delta H_r}$, are calculated using Equation 6.

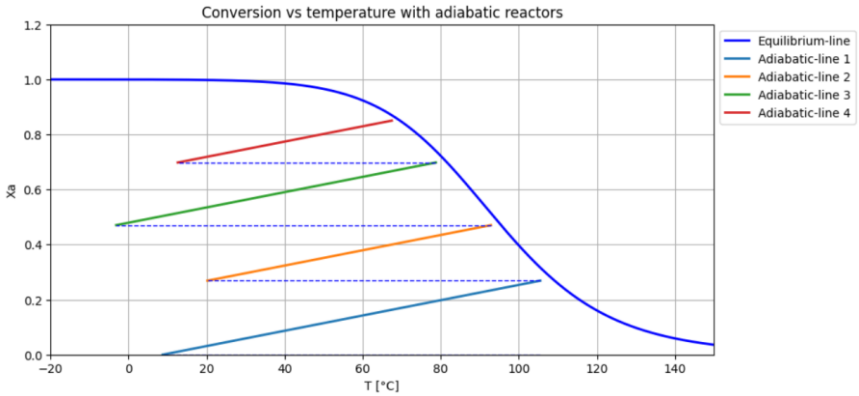


Figure 7: Example of an exothermic adiabatic configuration with intercooling

As shown in Figure 7, the adiabatic steps typically terminate before reaching the equilibrium line. This is an engineering decision, as the reaction rate becomes very low near the equilibrium line and eventually reaches zero, making further conversion inefficient within a single reactor [1]. Continuing the reaction under such conditions would require a disproportionately large reactor volume for only a small gain in conversion.

Therefore, to design an efficient reactor system, the adiabatic step is deliberately stopped at a point before the equilibrium is reached which is often based on a target minimum reaction rate, a maximum allowable reactor volume or a percentage of the maximum conversion. This strategy allows the system to bypass the low-rate region near equilibrium and achieve higher overall conversion using multiple adiabatic stages.

2.3 Levenspiel Plot

This section outlines the calculation of reactor volume for a PFR and a CSTR using the adiabatic lines in the X_A vs. T diagram. It begins with an explanation of the Levenspiel Plot, which plots the inverse reaction rate as a function of conversion [8]. Next, the reactor volumes for different reactor types are determined. Additionally, the required heat exchange between each reactor is calculated. Finally, strategies for optimizing the reactor configuration to minimize the total reactor volume are discussed.

2.3.1 Constructing inverse reaction rate vs. conversion chart

The Levenspiel plot is a graphical tool used to determine the required volume of chemical reactors, especially PFRs. In this plot, the inverse of the reaction rate, $\frac{1}{-r_A}$, is plotted against the conversion, X_A [8]. The surface area under the curve corresponds to the reactor volume required to reach a desired conversion under steady-state conditions [1, 8].

In the isothermal case, where the temperature is uniform across the entire reactor, the reaction rate depends only on the conversion [22]. The rate can be determined directly using the kinetic expression at that fixed temperature. Plotting $\frac{1}{-r_A}$ versus X_A then gives a smooth curve, and the area under this curve represents the required reactor volume.

However, in adiabatic reactors, the temperature increases or decreases with conversion depending on the reaction enthalpy. This change in temperature affects the reaction rate, since it becomes a function of both conversion and temperature [1]. In such cases, determining the rate at each conversion point requires additional attention.

The reaction rate in adiabatic systems can be estimated using analytical calculation. This involves computing the rate using a kinetic expression, such as Equation 5 in Section 2.1, which explicitly defines $-r_A(T, X_A)$. By calculating the rate at several points along the adiabatic path, a precise Levenspiel curve can be constructed as is shown in Figure 8.

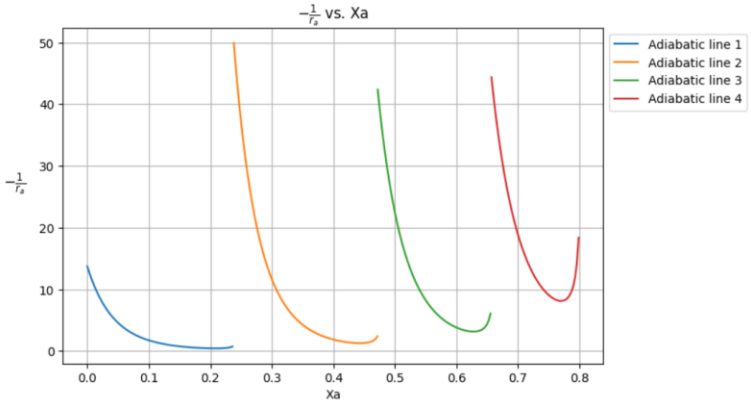


Figure 8: Levenspiel Plot for four adiabatic steps

2.3.2 Calculation of reactor volume

The calculation of the reactor volume depends on the type of reactor. Below, the methods for determining the reactor volume of a PFR and a CSTR are described.

Volume of a PFR

The reactor volume of a PFR can be determined using a Levenspiel plot. The fundamental equation for a PFR is derived from the differential molar balance, as shown in Equation 7 [23].

$$dV = \frac{F_{A0} \cdot dX_A}{-r_A} \tag{7}$$

In this equation, V represents the reactor volume, F_{A0} denotes the molar flow of component A, and $-r_A$ is the reaction rate. By integrating Equation 7 over the conversion range, Equation 8 is obtained [23].

$$\frac{V}{F_{A0}} = \int_{X_{A0}}^{X_{Af}} \frac{1}{-r_A}\, dX_A \tag{8}$$

This integral represents the area under the curve in the Levenspiel plot, as shown in Figure 9. A larger area corresponds to a larger reactor volume, as the reactant must remain in the reactor longer to achieve the desired conversion.

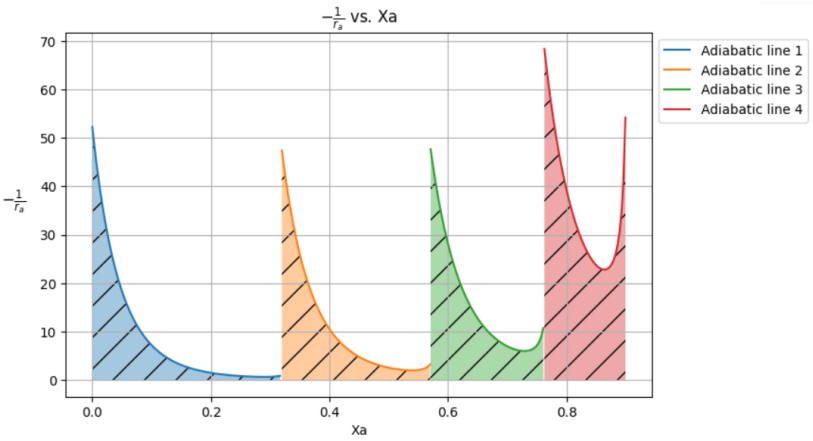


Figure 9: Levenspiel Plot for four adiabatic PFRs

Volume of a CSTR

The reactor volume of a CSTR can also be calculated using a Levenspiel plot. For a CSTR, the steady-state molar balance is given by Equation 9 [24].

$$V = \frac{F_{A0} \cdot (X_{Af} - X_{A0})}{-r_A} \tag{9}$$

Here, V represents the reactor volume, Fa_0 denotes the molar flow of component A and $-r_A$ is the reaction rate at the outlet conversion X_{Af}. Since a CSTR is perfectly mixed, the composition of the outlet stream is identical to that of the reactor contents [24]. This means that the reaction rate remains constant within the reactor and depends solely on the outlet concentration [24]. In a Levenspiel plot, a CSTR is represented as a rectangle with a width of $X_{Af} - X_{A0}$ and a height of $\frac{1}{-r_A}$. Figure 10 provides an example of the area corresponding to a CSTR in the Levenspiel plot.

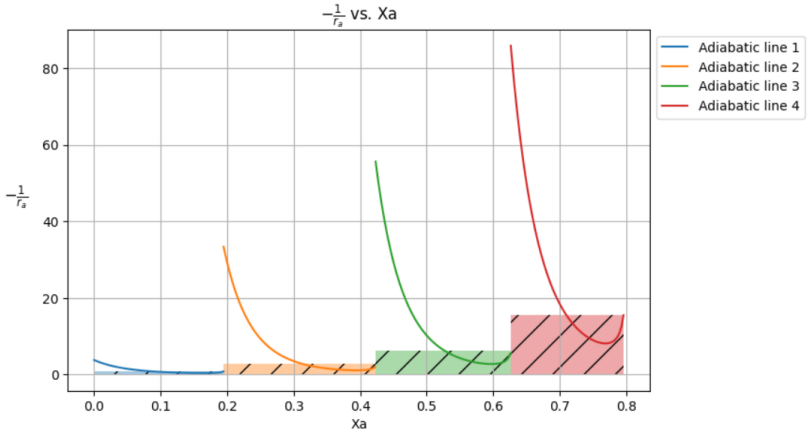


Figure 10: Surface area for four CSTRs

2.3.3 Determining the amount of intermediate heat exchange

To prevent an exothermic reaction from stopping due to reaching the equilibrium line, intermediate cooling is applied between each reactor as is discussed in Section 2.2.3. The amount of heat exchanged can be determined using Equation 10 [1].

$$Q = \dot{n} \cdot \bar{c_p} \cdot \Delta T \qquad (10)$$

In this equation, Q represents the amount of heat removed (J/s), $\dot{n}$ denotes the molar flow rate of the reaction mixture (mol/s), $\bar{c_p}$ represents the molar heat capacity (J/mol·K), and ΔT is the temperature difference (K) of the reaction mixture before and after cooling.

2.3.4 Optimizing the total reactor volume

This section discusses two methods to reduce the area in the Levenspiel plot and, consequently, the total reactor volume. These methods are introducing a recycle loop for a PFR and combining a CSTR with a PFR for a single adiabatic step.

PFR with a recycle loop

In certain cases, the required reactor volume of a PFR can be significantly reduced by implementing a recycle loop [1, 21]. As is explained in Section 2.2.1, the recycle loop helps to moderate the concentration and temperature profiles within the reactor, making the system behave more like a CSTR. This can enhance conversion efficiency and reduce the overall reactor size.

To determine the required reactor volume and the recycle ratio, a graphical method based on the Levenspiel plot can be used, as shown in Figure 11. This method is especially convenient for first-order reactions, where the $\frac{1}{-r_A}$ vs X_A curve allows a straightforward graphical construction to determine optimal recycle conditions [1, 21]. In these cases, the recycle behavior can be graphically approximated by finding a horizontal line on the plot such that the area above the adiabatic curve and below this line (green-shaded area) is equal to the area below the adiabatic curve and above this line (purple-shaded area) [1].

The point at which the horizontal line first intersects the adiabatic curve from the left corresponds to the conversion of the mixed feed entering the reactor, denoted as $X_{A,recycle}$ [1, 21]. This is not a location inside the reactor where the recycle starts, but rather the effective conversion of the inlet stream resulting from mixing the fresh feed ($X_A = 0$) with the recycled outlet stream. The final conversion at the reactor outlet is also denoted as X_A.

The recycle ratio R, which defines the amount of material being recycled relative to the amount exiting as product, can then be calculated using Equation 11 [1].

$$R = \frac{X_{A,recycle}}{X_A - X_{A,recycle}} \qquad (11)$$

In this equation, R represents the recycle ratio, $X_{A,\,recycle}$ is the conversion at the beginning of the reactor where the feed stream and the recycle stream are mixed, and X_A is the final conversion of the reactor.

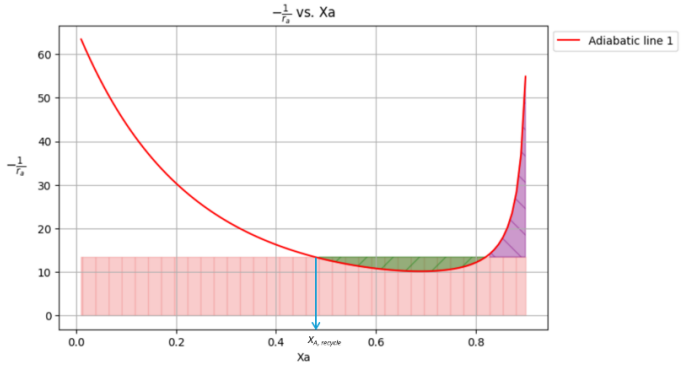


Figure 11: Surface area for one PFR with recycle

The reactor volume for a PFR with a recycle loop can be determined using the Levenspiel plot by calculating the area under the curve. The reaction rate throughout the reactor corresponds to the height of the horizontal line that intersects the adiabatic curve, as shown in Figure 11 [1].

The reactor volume V is then calculated by multiplying this constant height by the total span of conversion as shown in Equation 12 [1].

$$\frac{V}{F_{A0}} = \frac{1}{-r_A} \cdot (X_A - X_{A0}) \qquad (12)$$

In Figure 11, this area is represented as the sum of the red and green areas, which together form a rectangle under the horizontal line from $X_{A,0}$ to X_A [1]. This graphical method gives a straightforward way to estimate the reactor volume once the recycle ratio has been established.

Combination of CSTR and PFR

To maximize production efficiency and minimize the total reactor volume, it is important to reduce the area under the curve in the Levenspiel plot.

An effective strategy to achieve this is by combining CSTRs and PFRs as illustrated in Figure 12. This configuration takes advantage of the strengths of each reactor type at different points along the reaction pathway. Starting with a CSTR allows the process to immediately operate at a high reaction rate by jumping to a higher conversion [1]. This avoids the inefficiency of a PFR operating at low conversion, where the reaction rate is low and the required volume is large. Once a higher conversion is reached, the process continues in a PFR, which is more efficient at handling decreasing reaction rates as conversion increases [1]. In this way, the combination of a CSTR followed by a PFR aligns the reactor operation more closely with the kinetics of the reaction, reducing the total area under the curve in the Levenspiel plot, and thus minimizing total reactor volume and associated costs.

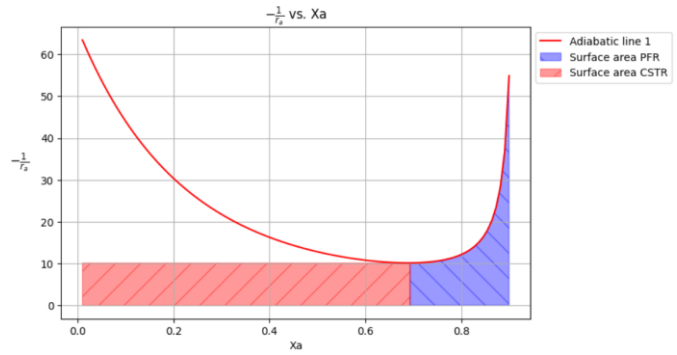


Figure 12: Surface area for combination CSTR and PFR for one adiabatic step

2.4 Genetic algorithm

Genetic algorithms (GA) are a family of heuristic optimization techniques inspired by natural evolution [25, 26, 27]. They are used to find solutions for complex problems where traditional methods may not always be effective. The algorithm generates a population of potential solutions and iteratively improving them using mechanisms such as selection, crossover, and mutation, as illustrated in Figure 13. In this research, two genetic algorithms are applied sequentially: a multi-objective genetic algorithm to optimize the adiabatic step configuration, followed by a single-objective genetic algorithm that selects the most suitable reactor configuration based on the optimal result from the first algorithm.

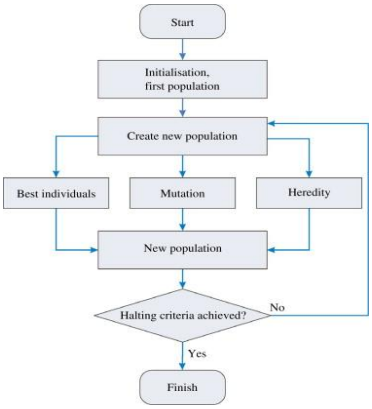


Figure 13: Basic structure of a genetic optimization algorithm [27]

Initialization phase

In the initialization phase, an initial population of possible solutions is randomly generated [27, 28]. Each solution is referred to as an individual, which is composed of one or more chromosomes. These chromosomes represent the decision variables of the problem and are typically encoded as strings of binary values or real numbers, depending on the application [27, 28]. The population size is chosen based on the complexity of the optimization problem and the available computational resources.

Once the initial population has been generated, each individual is evaluated using a fitness function that quantifies how well it solves the optimization problem [27, 28]. The evolutionary process begins with crossover and mutation mechanisms, which generate new individuals by recombining the genetic material of two parent solutions and applying small random modifications to maintain diversity [27, 28].

Crossover strategies

In this research, crossover is implemented using blend crossover (cxBlend) and two-point crossover (cxTwoPoint). Blend crossover is used in the multi-objective optimization part of the study, while two-point crossover is applied in the single-objective optimization part.

Blend crossover generates offspring by sampling new solutions from an extended range between two parent individuals [29, 30]. Say P_1 and P_2 the parent individuals, a child is created by selecting, for each gene, a value from a continuous interval that spans beyond the direct space between the two parents [29, 30]. This interval is illustrated in Figure 14 and consists of three regions: an exploitation zone, covering the direct range between the two parents and two exploration zones on either side, whose width is controlled by a blending parameter α [29, 30].

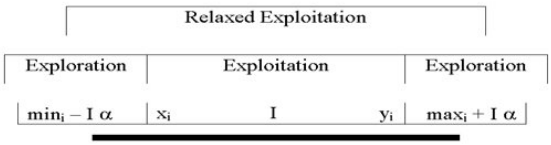


Figure 14: Blend crossover (cxBlend) [30]

As shown in Figure 14, this results in a relaxed exploitation interval, promoting both refinement and variability in the generated offspring [30]. This approach allows the algorithm to investigate a wider range of potential solutions while still focusing around the most promising areas discovered so far [30].

In the single-objective optimization algorithm, two-point crossover is used instead. In this method, two crossover points are randomly selected along the chromosome (individual), and the segments between these points are exchanged between two parent individuals [3, 4]. This operation creates two offspring that inherit parts from both parents, potentially combining their strengths. Two-point crossover is especially effective when dealing with ordered data structures, such as the sequence of reactor types in this research, and helps preserve useful substructures while still introducing variation [3, 4].

Parent selection for crossover also differs between the two algorithms. In the multi-objective optimization, NSGA-II is used, which applies an elitist selection strategy: high-performing individuals have a higher probability of being selected to reproduce [6, 7]. In contrast, in the single-objective optimization, tournament selection is employed. During tournament selection, individuals are randomly drawn from the population, and the fittest individual within this group is chosen to serve as a parent [6, 7, 28]. These two selection strategies are further discussed later in this section.

Mutation strategies

Mutation is a key genetic operator used to preserve diversity within the population and to prevent early convergence to local optima, as illustrated in Figure 15 [31, 32].

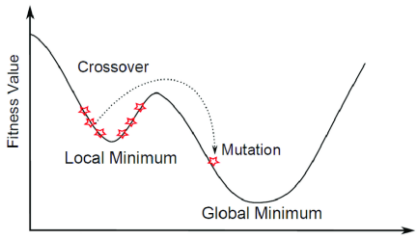


Figure 15: Mutation prevents getting stuck in a local minimum [31]

In this study, each chromosome represents a possible reactor configuration and is encoded as a real-valued vector or an ordered list, depending on the optimization phase. In the multi-objective optimization phase, each gene within the chromosome corresponds to the starting temperature of an individual adiabatic step in a sequential adiabatic system. These temperature values are normalized to lie within the range [0, 1], and are later scaled to the actual temperature bounds, for example from 0 to 100 degrees Celsius, during evaluation. Although other reactor design choices such as the number of reactors can also be optimization variables, this section focuses specifically on the optimization of starting temperatures, which serve as an important role in controlling the temperature trajectory and conversion behavior in adiabatic systems.

In the multi-objective optimization, gaussian mutation is applied to introduce variation into the population and avoid premature convergence. In the case of real-valued encoding, Gaussian mutation is employed: a single gene (i.e., a starting temperature value) is randomly selected from the chromosome, and a random perturbation is applied to it [31, 32]. This perturbation is drawn from a standard normal distribution, denoted as $r_2 \sim N(0,1)$. The current value of the gene, referred to as current, is then updated by adding r_2, as shown in Equation 13 [32].

$$new\ value = \max(0, \min(1, current + r_2)) \quad (13)$$

This step ensures that the mutated gene remains within the valid normalized range [0, 1], thereby respecting the predefined physical limits of the temperature domain. If the mutation causes the value to exceed 1 or drop below 0, it is set to the nearest bound [32].

In the single-objective optimization phase, the chromosome structure changes. Each individual encodes a sequence of reactor types (such as CSTRs, PFRs, or PFRs with recycle) rather than real-valued temperatures. Therefore, a different mutation method is used: mutation by shuffling indexes (mutShuffleIndexes). In this approach, the order of reactor types within the chromosome is randomly shuffled, either partially or entirely. This method is particularly effective for maintaining the diversity of combinatorial solutions, as it allows the algorithm to explore new reactor type sequences while preserving the overall set of available reactor types.

Fitness value

Next, a fitness value is calculated for each individual, indicating how well the solution performs relative to the optimized goal. The fitness function is problem-dependent and can, for example, evaluate the cost, efficiency, or route length [25, 26, 27]. Since evolutionary algorithms are inherently minimization algorithms, the optimization function should be formulated accordingly [27]. This means that higher performance should correspond to lower fitness values. This inversion is necessary because most evolutionary algorithms, including DEAP, are formulated as minimization problems [27]. For example, maximizing efficiency can be reformulated as minimizing the negative efficiency. Individuals with lower fitness values are therefore considered better and have a higher chance of being selected for the next generation [27].

Selection phase

Based on their fitness values, individuals are selected to generate offspring for the next generation [25, 26, 27]. Several selection strategies exist, including fitness-proportionate selection, tournament selection, and elitism [25, 26, 27]. In this study, different selection strategies are applied depending on the optimization phase.

In the multi-objective optimization, selection is performed using the Non-dominated Sorting Genetic Algorithm II (NSGA-II) [6, 7]. NSGA-II ranks individuals based on Pareto dominance and applies crowding distance to maintain diversity within the population. It combines elitism, preserving the best solutions across generations, with diversity preservation, ensuring a balanced exploration of the solution space [6, 7]. A more detailed explanation of the NSGA-II procedure is provided in Sections 2.5 and 2.6.

In the single-objective optimization, selection is performed using tournament selection. In this method, a random subset of individuals is selected from the population, and the best individual within this subset, the one with the lowest fitness value, is chosen as a parent [6, 7, 28]. This strategy encourages strong individuals to be selected while maintaining sufficient diversity to prevent premature convergence.

Termination phase

The algorithm terminates when a predefined stopping criterion is met [27]. This can include reaching a maximum number of generations, observing minimal improvement in fitness values over predefined number of iterations, or achieving a solution that satisfies a specific threshold or accuracy requirement [3].

2.5 Pareto-chart

The adiabatic configuration problem addressed in this thesis is a multi-objective optimization problem as discussed in Chapter 1, where multiple conflicting goals such as minimizing reactor volume, intercooling energy and the number of adiabatic steps must be optimized simultaneously [33, 34]. Unlike single-objective optimization, which targets a single optimal outcome, multi-objective optimization strives to find a range of trade-off solutions that balance conflicting objectives [34]. Genetic algorithms are highly effective for this task as they can investigate large and complex search spaces while maintaining population diversity [33, 34].

A Pareto chart is commonly used to visualize the outcome of such optimizations [33, 34]. It highlights the Pareto front which consists of non-dominated solutions for which no other solution performs better in all objectives simultaneously [33, 34]. These solutions represent optimal tradeoffs where improving one objective necessarily worsens another. For example, when minimizing cost and maximizing efficiency, a non-dominated solution has no alternative that is both cheaper and more efficient. Solutions that perform worse in all objectives are dominated and are excluded from the final selection.

As shown in Figure 16, the Pareto chart also illustrates feasible and infeasible regions. Feasible solutions satisfy all constraints and may contribute to the Pareto front [33]. Infeasible solutions violate at least one constraint and are discarded [33]. The goal is to evolve and retain high quality feasible non dominated solutions that enable informed decision making based on tradeoffs between objectives [33].

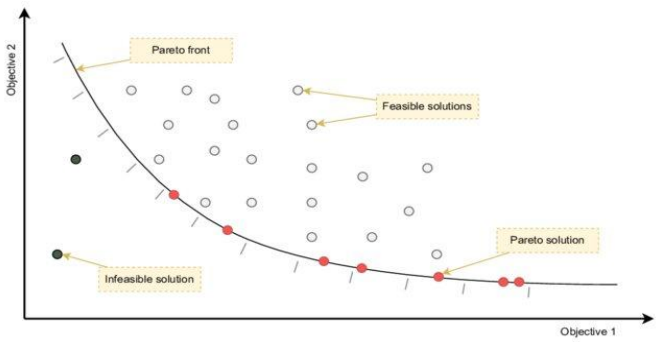


Figure 16: Pareto front for multi-objective optimization [33]

2.6 NSGA-II

NSGA-II (Non-dominated Sorting Genetic Algorithm II) is an evolutionary algorithm designed to solve multi-objective optimization problems by balancing convergence and diversity in the solution space [35, 36]. It preserves a population of candidate solutions that progressively evolve across generations by means of selection, crossover, and mutation [35, 36].

2.6.1 Non-dominated Sorting

A core step in NSGA II is non-dominated sorting, where solutions are ranked according to Pareto dominance [35, 36]. A Solution A is deemed superior to another solution B if it meets or exceeds B's performance across all objectives and strictly outperforms in at least one objective [35, 36]. Based on this rule, the algorithm assigns all non-dominated solutions, meaning solutions that are not dominated by any other in the population, to the first rank, also called Pareto front 1 [35, 36]. These solutions represent the best trade-offs currently found, as no other solutions perform better in all objectives at once. After identifying this front, the algorithm temporarily removes these solutions and repeats the process on the remaining ones to form Pareto front 2, and so on, as shown in Figure 17 [36]. An example of this is minimizing both cost and emissions in an industrial process. A solution that costs 80 euro and emits 45 kilograms of CO_2 dominates another that costs 95 euro and emits 55 kilograms of CO_2. However, a solution that costs 70 euro but emits 65 kilograms of CO_2 is non-dominated relative to one that costs 90 euro and emits 35 kilograms of CO_2, since neither performs better in both objectives.

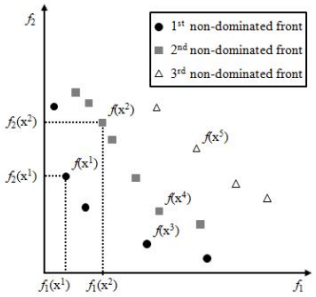


Figure 17: Non-dominated sorting of NSGAII [36]

2.6.2 Crowding Distance

To ensure diversity within a front, NSGA-II calculates a crowding distance for each solution. This is a measure of how isolated a solution is from its neighbors in the objective space [36, 37]. For each objective, the solutions are sorted, and boundary solutions receive an infinite crowding distance to always be retained [36, 37]. A distance is attributed to interior solutions by comparing the normalized difference in objective values with their nearest neighbors as is shown in Figure 18 [36, 37]. For instance, consider three solutions located on the same Pareto front with comparable costs but differing emission levels of 20 kg, 30 kg, and 50 kg. The intermediate solution (30 kg) resides in a more densely populated region of the objective space and consequently has a lower crowding distance. NSGA-II prioritizes the boundary solutions (20 kg and 50 kg) to preserve diversity across the front by favoring individuals that are more isolated.

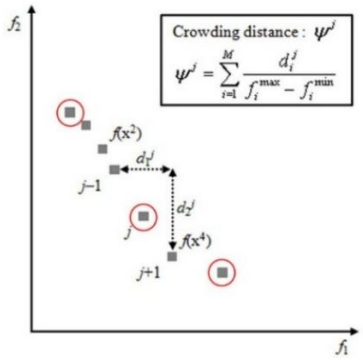


Figure 18: Determining crowding distance [36]

2.6.3 Selection

Selection in NSGA-II combines non-dominated sorting and crowding distance sorting as illustrated in Figure 19. When constructing the next generation, solutions from lower-rank (better) fronts are chosen first [37, 38, 39]. When a front contains more solutions than can be accommodated in the next generation, selection is based on crowding distance, favoring individuals located in less densely populated regions of the objective space to preserve solution diversity [37, 38, 39]. This strategy ensures that NSGA-II preserves the best trade-off solutions while maintaining a wide spread across the Pareto front [38, 39]. Over successive generations, the population converges towards the true Pareto front with improved diversity [38, 39].

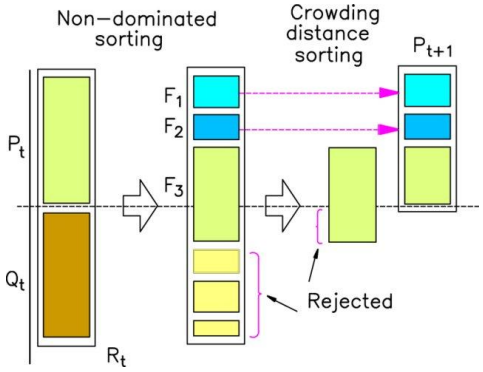


Figure 19: Selection process of NSGAII [39]

3. Case description and reactor optimization methodology

This chapter presents the methodology employed throughout this thesis. The chapter begins with the formulation of the reaction kinetics that forms the basis of the simulation framework (Section 3.1). Subsequently, Sections 3.2 and 3.3 outline how two standard genetic algorithms were implemented and adapted for the optimization of reactor configurations. The first algorithm is a multi-objective optimization algorithm designed to determine the optimal sequence of adiabatic steps by simultaneously minimizing the surface area under the Levenspiel plot, the amount of intercooling, and the total number of adiabatic steps (Section 3.2). Building upon the results of the first, the second algorithm is a single-objective optimization algorithm that selects the most suitable reactor types (CSTR, PFR, Hybrid and PFR with recycle) to minimize the overall reactor volume required for the desired conversion (Section 3.3). Then the method to assess the performance of the two-step algorithm is described (Section 3.4). Finally, the materials used in this thesis are introduced (Section 3.5).

3.1. Calculation of reaction Kinetics

In this thesis, four different case studies were analyzed. Two of these are based on examples from the work of Levenspiel [1, pp. 218 and 437], the third is the WGSR and the fourth concerns a hypothetical reaction system designed to test the flexibility of the algorithm. As an example, this section describes the calculated reaction kinetics of the WGSR. The other cases are further explained in Appendix I.

The equilibrium curve is established using the standard enthalpies of formation and standard Gibbs free energies of the reactants and products. The reaction equation for the WGSR is presented in the following expression [10].

$$CO(g) + H_2O(g) \rightleftharpoons CO_2(g) + H_2(g)$$

To calculate $\Delta H°$, the sum of the standard enthalpies of formation of the reactants is subtracted from that of the products, as shown below [40].

$$\Delta H° = -393.5\ (CO_2) + 0\ (H_2) - [(-110.5)\ (CO) + (-241.8)\ (H_2O)] = -41.2\ kJ/mol$$

Similarly, $\Delta G°$ is determined using the standard Gibbs energies of formation [40].

$$\Delta G° = -394.4\ (CO_2) + 0\ (H_2) - [(-137.2)\ (CO) + (-228.6)\ (H_2O)] = -28.6\ kJ/mol$$

Using the calculated values for $\Delta H°$ and $\Delta G°$, the equilibrium constant at 298 K and its temperature dependence, can be determined. This constant is obtained using Equation 1 (Section 2.1), as demonstrated in the equation below.

$$K_{298} = e^{\frac{-(-28600)}{8.314 \cdot 298}} = 103111$$

Based on this equilibrium constant at 298 K, the temperature dependence of the equilibrium constant can be derived using the van 't Hoff equation (Equation 2, Section 2.1), enabling construction of the equilibrium curve as a function of temperature and conversion.

$$K = 103111 \cdot e^{\frac{-(-41200)}{R} \cdot \left(\frac{1}{T} - \frac{1}{298}\right)} = e^{\frac{41200}{R \cdot T} - 5{,}09}$$

The discussion so far has focused on the thermodynamic aspects of the WGSR. From this point on, the kinetic behavior of the reaction is addressed. From literature, the values for the activation energy $E_{a,1}$ (i.e., the minimum energy required for the forward reaction to proceed) and the pre-exponential factor $ln(k_0)$ of the Arrhenius equation (Equation 4, Section 2.1) were obtained [41]. Substituting these values into the Arrhenius equation yields the following expression for the temperature-dependent rate constant of the forward reaction of WGSR.

$$k_1 = e^{12.60 - \frac{47400}{R \cdot T}}$$

Using the expressions for the equilibrium constant and the forward reaction rate constant, the temperature-dependent rate constant of the reverse reaction k_2 can be derived. This is calculated using Equation 3 (Section 2.1), as is illustrated beneath.

$$k_2 = \frac{e^{12.60 - \frac{47400}{R \cdot T}}}{e^{\frac{41200}{R \cdot T} - 5.09}} = e^{17.69 - \frac{10656.7}{T}}$$

In order to account for the effect of volumetric changes during the reaction, the expansion factor ε was evaluated. This dimensionless parameter is defined as the relative change in the number of moles of gas per mole of reactant gas. The total number of moles of gas remains constant (2 mol of gas reactants and 2 mol of gas products), resulting in the following equation.

$$\varepsilon = \frac{\Delta n}{n_{reactants}} = \frac{2 - 2}{2} = 0$$

The volumetric flow rate is assumed constant throughout the reactor. Although the reaction is exothermic and adiabatic, and thus the temperature increases, the effect of temperature-induced expansion is neglected for simplicity. In reality, this would lead to a moderate increase in flow rate under constant pressure.

To plot the reaction rates in the conversion versus temperature diagram, the conversion can be calculated as a function of temperature at various reaction rates. The equation below shows how the conversion is determined.

$$X_A = \frac{k_1 \cdot C_{A0} - r_A}{k_1 \cdot C_{A0} + k_2 \cdot C_{A0}}$$

Using the derived kinetic expressions for the water-gas shift reaction, the X_A vs. T-diagram was subsequently constructed, as presented in Figure 20.

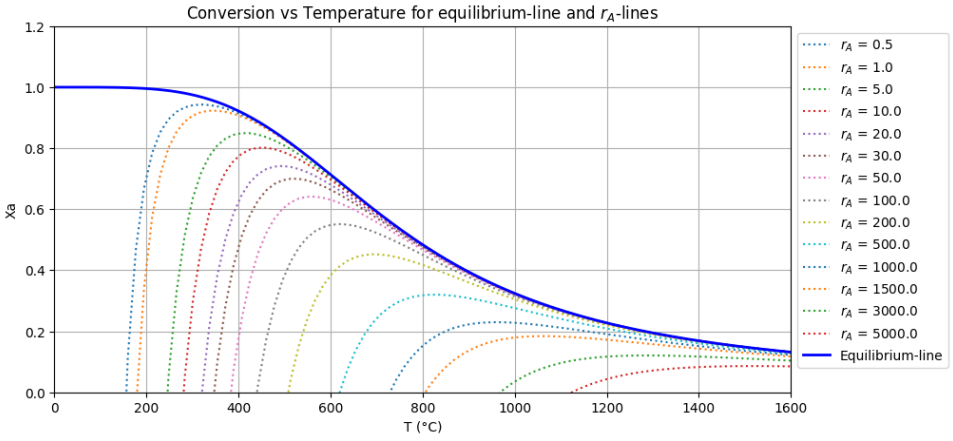


Figure 20: Equilibrium line and reaction rates for WGSR

3.2. Identification of the optimal adiabatic step temperatures

This section describes the multi-objective optimization implementing the NSGA-II algorithm for identifying the optimal inlet temperatures of each adiabatic step, using the reaction kinetics for each case study determined in Section 3.1. The goal is to minimize three key performance objectives, namely the total surface area under the adiabatic curve in the Levenspiel plot as discussed in Sections 2.3.1 and 2.3.2, the total amount of heat exchange required for intermediate cooling between the adiabatic steps using Equation 10, and the number of adiabatic stages in the configuration, while simultaneously achieving the desired conversion of 0.95. The objective weights that are used to evaluate the performance of the algorithm in this thesis are further explained in Section 3.4.

The algorithm iteratively explores configurations for different adiabatic step counts, ranging from one up to a predefined maximum (10 steps). For each configuration, a population of candidate solutions is generated and evaluated. The DEAP 1.4.2 framework is used to handle evolutionary operations, including crossover, mutation, and selection based on the NSGA-II principles. A Pareto front is maintained for each configuration, capturing the non-dominated solutions. Across all configurations, the best overall solution is selected based on the combined score of the objectives.

3.2.1. Initialization

Each temperature value, or gene, is randomly initialized using a uniform distribution within a predefined temperature range that ensures physical and chemical feasibility. The lower bound is fixed at 20 °C, representing the minimum practical operating temperature for most industrial setups. The upper bound is fixed at a specific temperature, determined based on the equilibrium line for each case study. This temperature corresponds to a point on the equilibrium curve where the achievable conversion remains below 0.2. As a result, excessively high inlet temperatures that would otherwise result in very low conversions are avoided.

Once the bounds are defined, the DEAP framework is used to construct the genetic representation of the individuals. A custom generator function samples random starting temperatures for each adiabatic step using a uniform distribution within the defined bounds. This ensures a wide initial diversity in the population and avoids bias toward any specific configuration.

Each individual has a multi-objective fitness to evaluate several criteria at once. In this study, the three fitness objectives are the total area under the $\frac{1}{-r_A}$ vs. X_A curve which indicates reactor size, the amount of heat exchange required between adiabatic steps and the total number of adiabatic steps used in the configuration.

These objectives are inherently to be minimized. Since DEAP's default evolutionary algorithm is based on maximization, negative fitness weights are applied to convert the problem into a minimization task. Specifically, the FitnessMulti class is defined with three negative weights, -1.0 for each objective, to ensure proper ranking during the selection phase.

After defining the individual structure and fitness criteria, the population is initialized. A predefined number of individuals are generated, each with a random but valid set of starting temperatures. In this study, a population size of 100 individuals is used, and the algorithm is evolved over 150 generations. These values were selected to ensure sufficient exploration of the solution space while preserving an acceptable computational cost. The values can be adjusted depending on the desired trade-off between optimization quality and computation time.

3.2.2. Evaluation

The evaluation process of the optimization algorithm is based on a multi-objective assessment of each individual solution, representing a list of initial temperatures for each adiabatic step, e.g., [T_1, T_2, ..., T_n]. This evaluation aims to optimize the three primary objectives discussed in Section 3.2.1.

The evaluation begins by computing the conversion achieved by each adiabatic step using a mathematical function. This function estimates how far a step can progress along its adiabatic trajectory before it reaches the thermodynamic equilibrium. This final temperature of each adiabatic step is adjusted by multiplying the total temperature rise of the step by a predefined factor of 0.96, ensuring the reaction stops before reaching equilibrium and avoids the low-rate region where further conversion would require excessive reactor volume, as discussed in Section 2.2.3. If the conversion for an adiabatic step is too small, the trajectory is extended until the minimum conversion (predefined as 0.05) is reached.

Once the final temperatures and conversions of all steps are calculated, a function is used to reconstruct each adiabatic path as a series of tuples (X_A, $-r_A$, $1/-r_A$), where the reaction rate is determined using Arrhenius kinetics (Equation 4, Section 2.1).

Using the calculated reaction rates of the adiabatic steps, the surface area under the $1/-r_A$ vs. X_A curve for each step is determined using the trapezoidal rule, and the total area is summed across all steps. This value is used as a proxy for total reactor volume.

Next, another function determines the thermal energy required to bring the feed or reactor effluent to the next adiabatic step's start temperature. The first cooling or heating duty is computed from the initial feed temperature (defined as 20 °C) to the starting temperature of the first adiabatic step, and subsequent intercooling values are calculated between the following stages. The amount of heat exchange is determined using Equation 10 of Section 2.3.3.

In addition to these physical objectives, a penalty term is introduced into the fitness value of the surface area objective if the final conversion differs from the target conversion defined as 0.95. The penalty is proportional to the deviation $|X_{A,final} - 0.95| \times 10^4$. Furthermore, if any adiabatic step achieves a conversion step smaller than 0.05, an additional penalty of 1000 is added per violation to discourage underperforming stages.

Finally, the three objectives are scaled to ensure a fair comparison during optimization. This prevents any single objective from disproportionately influencing the selection process due to differences in magnitude or units. Specifically, fixed scaling factors are used to bring the values of these objectives into a comparable numerical range without affecting the shape of the Pareto front. The number of adiabatic steps is multiplied by a factor of 5, while the surface area and intercooling duty objectives are scaled differently depending on the case. These scaling factors are determined in Section 4.1. By bringing all objectives onto a similar scale, the algorithm can more effectively balance trade-offs and ensure that no single objective dominates the selection process due to its unit size or numerical range.

The three objectives are then passed to the NSGA-II algorithm for Pareto-based multi-objective optimization. For each configuration, ranging from 1 up to a predefined maximum of 10 adiabatic steps, the algorithm evolves a population over 150 generations to identify optimal trade-offs between the three objectives.

3.2.3. Breeding

The breeding process in the NSGA-II algorithm determines how new candidate solutions (offspring) are generated from existing individuals (parents). It makes use of genetic operators such as crossover and mutation to introduce diversity and explore the search space as discussed in Section 2.4. In each generation, a new set of individuals is created through recombination and variation, after which they are evaluated and selected based on their fitness.

Probabilities

To maintain a balance between exploration and exploitation during the genetic search, the algorithm applies a crossover probability of 0.5 and a mutation probability of 0.3. This means that 50% of the population will undergo crossover to produce offspring, while 30% will be subjected to mutation. These values help ensure sufficient diversity while preserving high-performing traits.

Crossover

The algorithm uses a blend crossover method (cxBlend) with an alpha parameter set to 0.3. This crossover technique generates offspring by combining the genes (inlet temperatures) of two parent individuals. For each gene, a value is randomly selected from an extended range defined by the minimum and maximum values of the corresponding genes in the parents, widened by a proportion determined by alpha [29, 30]. An alpha of 0.3 means that the sampling range extends 30% beyond both ends of the interval between the parent values. This encourages exploration of the space between and slightly beyond the parent values, promoting diversity in the offspring population.

Mutation

Mutation is implemented via the Gaussian mutation operator (mutGaussian). Each gene in an individual has a 30% probability (indpb = 0.3) of undergoing mutation by adding Gaussian noise, characterized by a mean (mu) of 50 and a standard deviation (sigma) of 20. This introduces controlled randomness to the solution, allowing the algorithm to explore new regions and escape potential local optima as discussed in Section 2.4.

3.2.4. Selection

The selection of individuals is performed using the selNSGA2 function from the DEAP library, which implements the NSGA-II algorithm. This method ranks individuals using non-dominated sorting and applies a crowding distance metric to preserve diversity within each Pareto front, described in Section 2.6.

A $(\mu + \lambda)$-strategy is employed in each generation, where the parent population (μ) and the offspring population (λ) both consist of 100 individuals. This means that in every generation, 100 new individuals are created through crossover and mutation, and these are combined with the 100 existing parents to form a temporary pool of 200 candidates. Based on their Pareto rank and crowding distance, the top 100 individuals from the pool are chosen to form the next generation.

This elitist strategy guarantees that the best-performing solutions are retained while maintaining diversity, allowing the algorithm to efficiently explore trade-offs between total surface area, cooling requirements, and adiabatic step count.

3.3. Identification of the optimal reactor type for each adiabatic step

In this section, the methodology of the second algorithm, which is employed to determine the optimal reactor configuration based on the best solution from the NSGA-II (focused on the adiabatic step configuration) discussed in Section 3.2, is outlined. This single-objective algorithm builds upon the results of the first NSGA-II and aims to select the most suitable reactor types, such as CSTR, PFR (or a combination of CSTR and PFR) or PFR with recycle, for each adiabatic step. The objective of the second algorithm is to minimize the overall reactor volume required to reach the desired conversion. This algorithm operates with a single fitness function, which evaluates each reactor configuration based on the total surface area under the Levenspiel Plot. The procedure for calculating the surface area for each type of reactor is discussed in Sections 2.3.2 and 2.3.4. Through evolutionary operators, the population of potential reactor configurations is iteratively improved, with the goal of finding the optimal reactor configuration that minimizes total surface area.

3.3.1. Initialization

The initialization step of the algorithm begins with the creation of a population of individuals, where each individual represents a possible configuration of reactor types assigned to a sequence of adiabatic reactor segments. To represent these configurations, each individual is encoded as a list of reactor options such as "PFR", "CSTR", "CSTR-PFR", or "PFR-recycle". These options are discussed in Section 2.3.

The DEAP library is used to set up the GA framework. A custom fitness class is defined to minimize the total surface area, the single objective of the optimization. The individual class inherits from Python's list structure, allowing it to store reactor configurations while also holding the associated fitness values.

The toolbox is then registered with functions to generate reactor type attributes (randomly selecting one of the four reactor options), create individuals (lists of reactor options), and build the population by repeating this process for a number of individuals (predefined as 50). This setup ensures that the GA can evolve and optimize the reactor configuration to minimize the total reactor volume.

The size of the initial population is set to 50, ensuring a diverse starting point for the evolutionary process. Each individual is initialized with a sequence of randomly assigned reactor options, matching the number of adiabatic reactor segments under consideration. This randomized initialization allows the algorithm to explore a wide variety of reactor combinations in the early generations, facilitating a good search for optimal configurations through subsequent evolutionary operations (crossover and mutation).

3.3.2. Evaluation

The evaluation process assesses the quality of reactor configurations generated by the genetic algorithm by calculating the total surface area under the Levenspiel plot for each proposed configuration. This includes different reactor options such as CSTR, PFR, CSTR-PFR combinations, and PFR with recycle, with the calculation method for each reactor option described in Section 2.3. The total surface area is used as a fitness value to evaluate each reactor configuration, with smaller surface areas indicating more efficient designs that require less reactor volume to reach the target conversion of 0.95. This fitness measure guides the selection process in the genetic algorithm, allowing it to favor and refine the most efficient configurations through successive generations.

3.3.3. Breeding

The breeding process starts by selecting two parent individuals from the population using a tournament selection method. During this process, a random subset of individuals is selected, and the best among them is chosen as a parent. The algorithm then generates offspring by employing crossover and mutation mechanisms to the selected parents.

Probabilities

The algorithm defines two probabilities to guide the genetic operations: the crossover probability (cxpb) and the mutation probability (mutpb). The crossover is assigned to 0.5, meaning there is a 50% chance for a crossover to occur. This operation promotes exploration by combining different parent configurations, enabling the discovery of new solutions. On the other hand, the mutation probability is defined as 0.2, indicating a 20% chance that an individual will undergo mutation. Together, these probabilities help balance exploration and exploitation within the algorithm, allowing it to efficiently explore the solution space while fine-tuning existing solutions.

Crossover

The crossover mechanism entails exchanging parts of two parent individuals to produce offspring. The algorithm applies two-point crossover (denoted as cxTwoPoint), where two crossover points are randomly chosen, and the portions between them are swapped between the parents [3, 4]. This creates two offspring that inherit traits from both parents, potentially combining their strengths [3, 4].

Mutation

The mutation process introduces minor random variations into an individual to maintain diversity within the population and reduce the risk of the algorithm becoming trapped in local optima. The mutation used in this algorithm is mutation by shuffling indexes (denoted as mutShuffleIndexes), where the order of reactor types in an individual is randomly shuffled. This change can help explore new configurations by slightly altering the existing solutions.

3.3.4. Selection

In the algorithm, selection determines which individuals will be passed on to the following generation. The selection process used in this algorithm is based on tournament selection. This mechanism randomly selects a small group of individuals from the population, and the individual with the best fitness is selected to reproduce [6, 7]. The tournament size, that refers to the number of individuals considered in each selection event, is set to three. This means that three individuals are randomly selected for each tournament, and the individual with the best fitness value, i.e., the individual that minimizes the total reactor surface area, is chosen to be part of the next generation. This mechanism is repeated until the desired number of individuals is chosen.

3.4 Performance evaluation of the two-step algorithm

To evaluate the performance of the two-step genetic algorithm, fifty-two simulations were conducted with systematic variation in the relative importance of the three optimization objectives described in Section 3.2.1. The precise calculation methods for these objectives are detailed in Section 3.2.2. Various combinations of weights are applied to the fitness values to test the algorithm's ability to generate meaningful and balanced reactor configurations under different design priorities. The evaluation starts with similar weights for all objectives as discussed in Section 3.2.2, after which each individual objective is increased in importance by a multiplication factor of 5, 10, 25, and 50, while keeping the others constant. This approach allows for assessing how the algorithm responds to shifting priorities and whether it adapts by generating a correspondingly logical and optimal adiabatic reactor configuration. For each combination, the resulting designs are examined with respect to conversion, reactor volume, intercooling needs, and the total number of adiabatic stages. The results provide insight into how changes in objective weighting influence the final reactor configuration.

3.5. Materials and cases

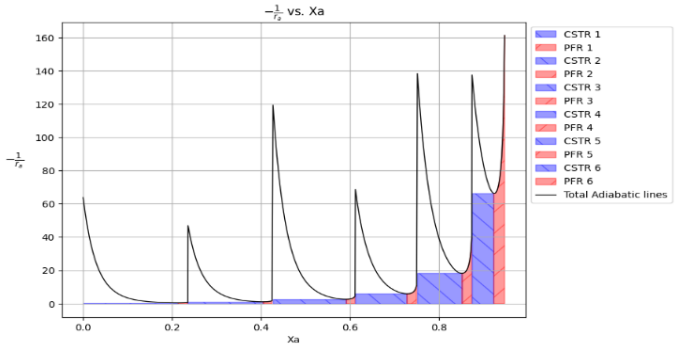
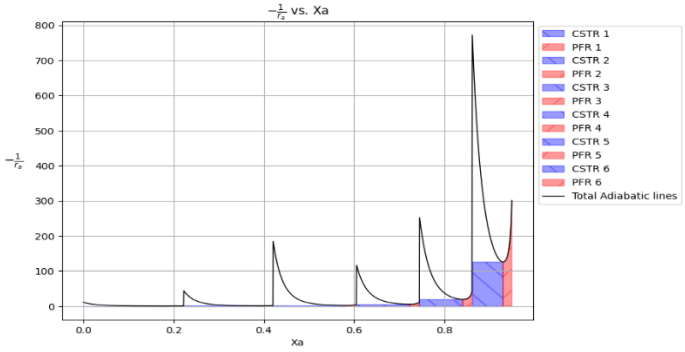
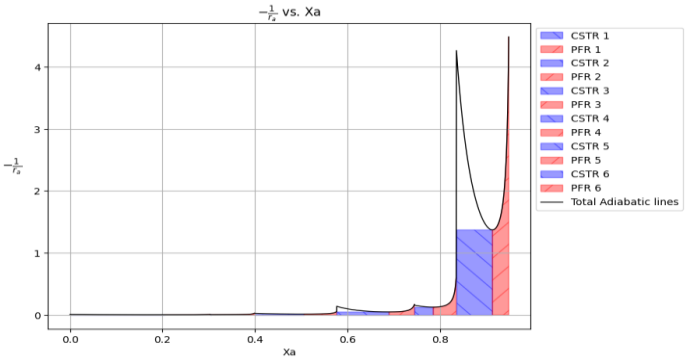
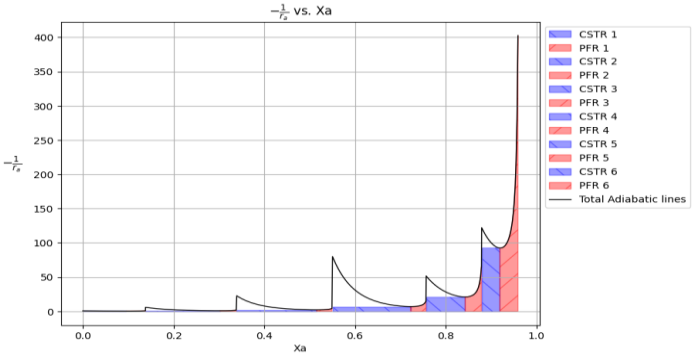
The computational framework for this research was developed using Python 3.10, with Google Colab as the primary environment, supported by a range of open-source libraries tailored for optimization and data visualization. The core of the optimization routine is built using DEAP 1.4.2 (Distributed Evolutionary Algorithms in Python), which provides tools for implementing evolutionary algorithms, including the multi-objective NSGA-II used in this thesis [42]. Numerical calculations were performed using NumPy and SciPy, enabling the integration of reaction rates, data interpolation, and equation solving. Matplotlib was employed for visualizing results, while Pandas was used to manage structured data and track population evolution during optimization.

For language support, ChatGPT and Grammarly are used to refine grammar, improve clarity, and enhance overall readability [43, 44]. While these tools assisted in improving the language and clarity of the text, all technical content, analysis, and conclusions are entirely my own.

To test and validate the developed compartmental optimization algorithm, four case studies were investigated. The first case, based on an example from Levenspiel, features a reversible, first-order, exothermic reaction $A \rightleftharpoons R$. This case served as an initial test for the algorithm, demonstrating the basic principles of compartmentalization and temperature optimization. The second case, also derived from Levenspiel, involves a similar reaction but with increased temperature sensitivity due to a higher activation energy. The third case study focuses on the industrially relevant water-gas shift reaction, and the final case is a hypothetical system designed to challenge the algorithm with an even higher activation energy than the other three cases. The two Levenspiel cases and hypothetical case are further explained in Appendix A.

Table 1 provides a summary of the four evaluated cases, including the expressions for both the forward and reverse reaction rate constants. Additionally, the optimal reactor configuration identified for each case is presented, focusing on scenarios with similar objective weights. The scaling factors applied to the three objectives in these scenarios are discussed in Section 4.1.

Table 1: Cases with optimal reactor configurations based on similar objective weights

Cases with rate expressions	Optimal reactor configuration
Case 1 (Levenspiel) $k_1 = e^{17.34-\frac{48900}{R\cdot T}}$ $k_2 = e^{42.04-\frac{124200}{R\cdot T}}$ $Cp = 40\ J/mol\cdot K$ $\Delta H_r = 18000\ J/mol$	
Case 2 (Levenspiel) $k_1 = e^{7-\frac{83700}{R\cdot T}}$ $k_2 = e^{18-\frac{167400}{R\cdot T}}$ $Cp = 50\ J/mol\cdot K$ $\Delta H_r = 80000\ J/mol$	
Case 3 (WGSR) $k_1 = e^{12.60-\frac{47400}{R\cdot T}}$ $k_2 = e^{17.69-\frac{10656.7}{T}}$ $Cp = 34.3\ J/mol\cdot K$ $\Delta H_r = 41200\ J/mol$	
Case 4 (Hypothetical) $k_1 = e^{17.73-\frac{83000}{R\cdot T}}$ $k_2 = e^{41.51-\frac{202600}{R\cdot T}}$ $Cp = 30\ J/mol\cdot K$ $\Delta H_r = 15000\ J/mol$	

4. Optimization Results Based on Weight Variations

This chapter presents the results from the simulations conducted using the two-step genetic algorithm discussed in Sections 3.2 and 3.3. The focus was on evaluating the performance of the algorithm while varying the weights of the fitness values for the three objectives described in Section 3.4. The algorithm was constrained to search for optimal configurations with up to 10 adiabatic steps, selecting the most suitable number between 1 and 10 for each case. In all simulations, the desired conversion was set to 0.95, and the algorithm aimed to achieve this target as efficiently as possible. By testing different weight combinations, the goal was to assess whether the algorithm adapts appropriately by generating adiabatic reactor configurations that align with the given design priorities such as minimizing reactor volume, reducing intercooling requirements, or limiting the number of adiabatic steps for each set of weights and case study. Additionally, the chapter discusses how variations in weights influenced the final reactor configurations.

4.1 Similar weight for all objectives

Before varying the weights of the fitness values for each objective, an initial optimal adiabatic reactor configuration was determined using fitness values that lie within the same order of magnitude. To achieve this, scaling factors were applied to bring the objectives to a comparable range. These initial scaling factors were obtained through a trial-and-error approach, in which the fitness values of the individual objectives were iteratively adjusted and evaluated until they were approximately of equal magnitude. The final scaling factors are listed in Table 2.

Table 2: Initial scaling factors for the three objectives (x = multiply, ÷ = divide)

Objective	**Scaling factor**			
	Case 1	**Case 2**	**Case 3**	**Case 4**
Surface area	No factor	÷ 2	x 90	x 2
Required intercooling	÷ 100	÷ 500	÷ 300	÷ 50
Number of stages	x 5	x 5	x 5	x 5

Using the scaling factors from Table 2, the two-step algorithm generated optimal reactor configurations for the different case studies. The calculated fitness values for the best configurations of the four cases are shown in Table 3.

Table 3: Fitness values for the three objectives using the initial scaling factors

Objective	**Fitness value**			
	Case 1	**Case 2**	**Case 3**	**Case 4**
Surface area	22.31	19.98	24.57	39.14
Required intercooling [J/s]	43.91	30.61	26.52	40.54
Number of stages	30.00	30.00	30.00	30.00

Figure 21 shows the conversion versus temperature profiles and the corresponding Levenspiel plots for the optimal adiabatic reactor configurations of each case, based on the initial scaling factors.

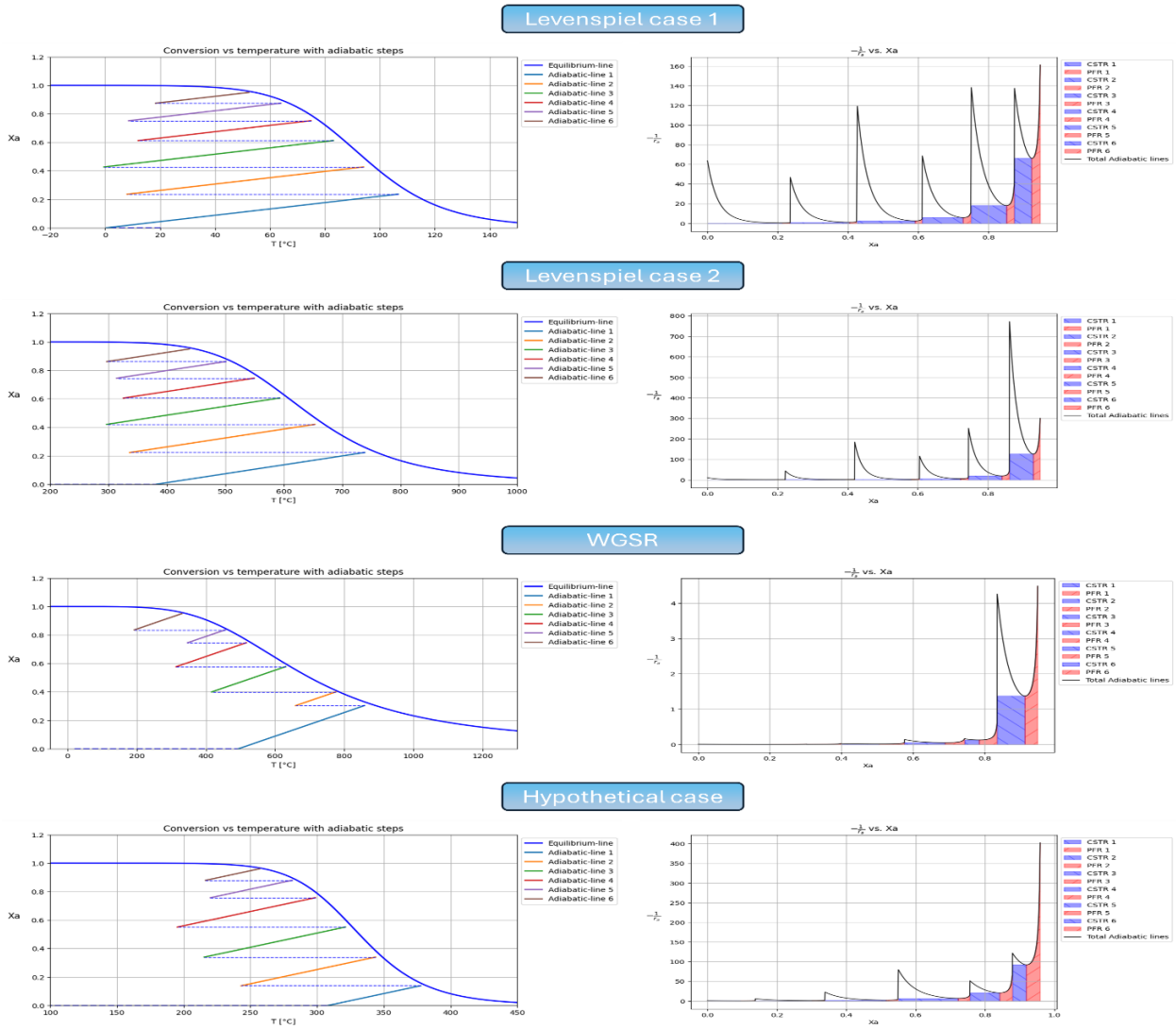


Figure 21: Optimal configurations for cases 1 to 4 (top to bottom) using initial scaling factors

Table 4 summarizes the objective values for the optimal configurations generated by the algorithm for each case study.

Table 4: Objective values for each case using the initial scaling factors

	Case 1	Case 2	Case 3	Case 4
Surface area	22.31	39.96	0.273	19.47
Required intercooling [J/s]	4391.3	15304.2	7956.3	1957.1
Number of stages	6	6	6	6

As shown in Figure 21 and Table 4, the algorithm consistently identified an optimal configuration consisting of six adiabatic steps for each case. This outcome indicates that, under balanced objective weighting, a six-step configuration represents an effective compromise between the three objectives. This result is both reasonable and expected. Segmenting the conversion trajectory into six steps allows the system to remain closer to the optimal temperature range for the reaction, thereby enhancing the average reaction rate and reducing the total reactor volume. At the same time, it avoids excessive

segmentation, which would increase the number of intercooling units and add unnecessary design complexity due to more adiabatic steps. Therefore, the selection of six steps reflects a balanced design strategy where no single objective dominates the optimization.

4.2 Optimization preference toward Levenspiel plot area

To research the influence of the surface area objective on the optimal reactor configuration, the weight associated with the fitness value for surface area was systematically increased while keeping the weights of the other two objectives constant. The scaling factor for each case used in Section 4.1 was increased with a multiplication factor of 5, 10, 25, and 50 relative to the other objectives. This approach evaluates how strongly prioritizing reactor volume reduction affects optimal configurations.

Figure 22 shows the optimal adiabatic reactor configurations obtained for each case when the surface area objective is scaled 5 times higher than the other two objectives.

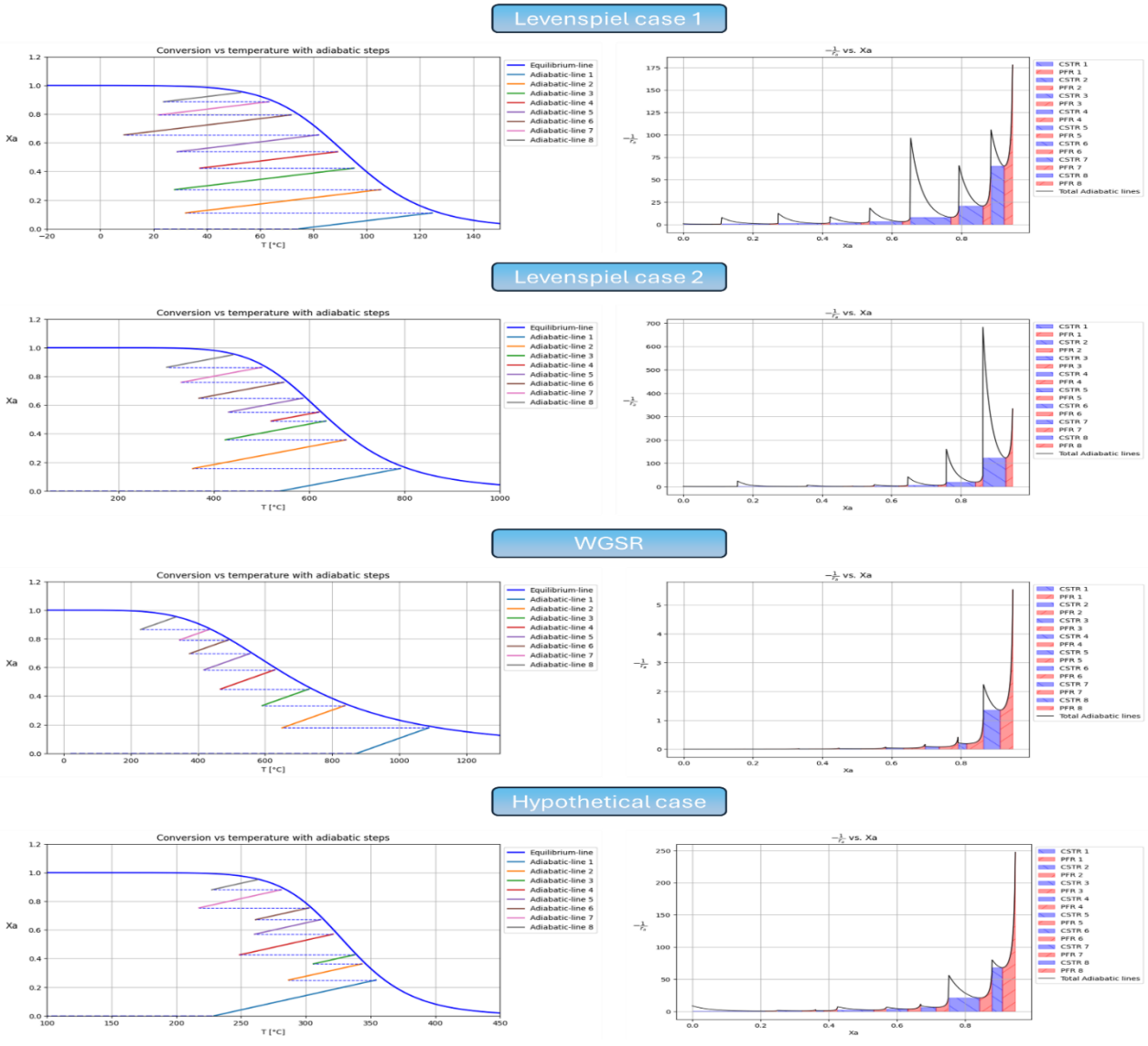


Figure 22: Optimal results based on a 5× scaling factor for surface area

Figure 23 shows the optimal configurations generated for all cases when the surface area objective is scaled 10 times higher than the other two objectives.

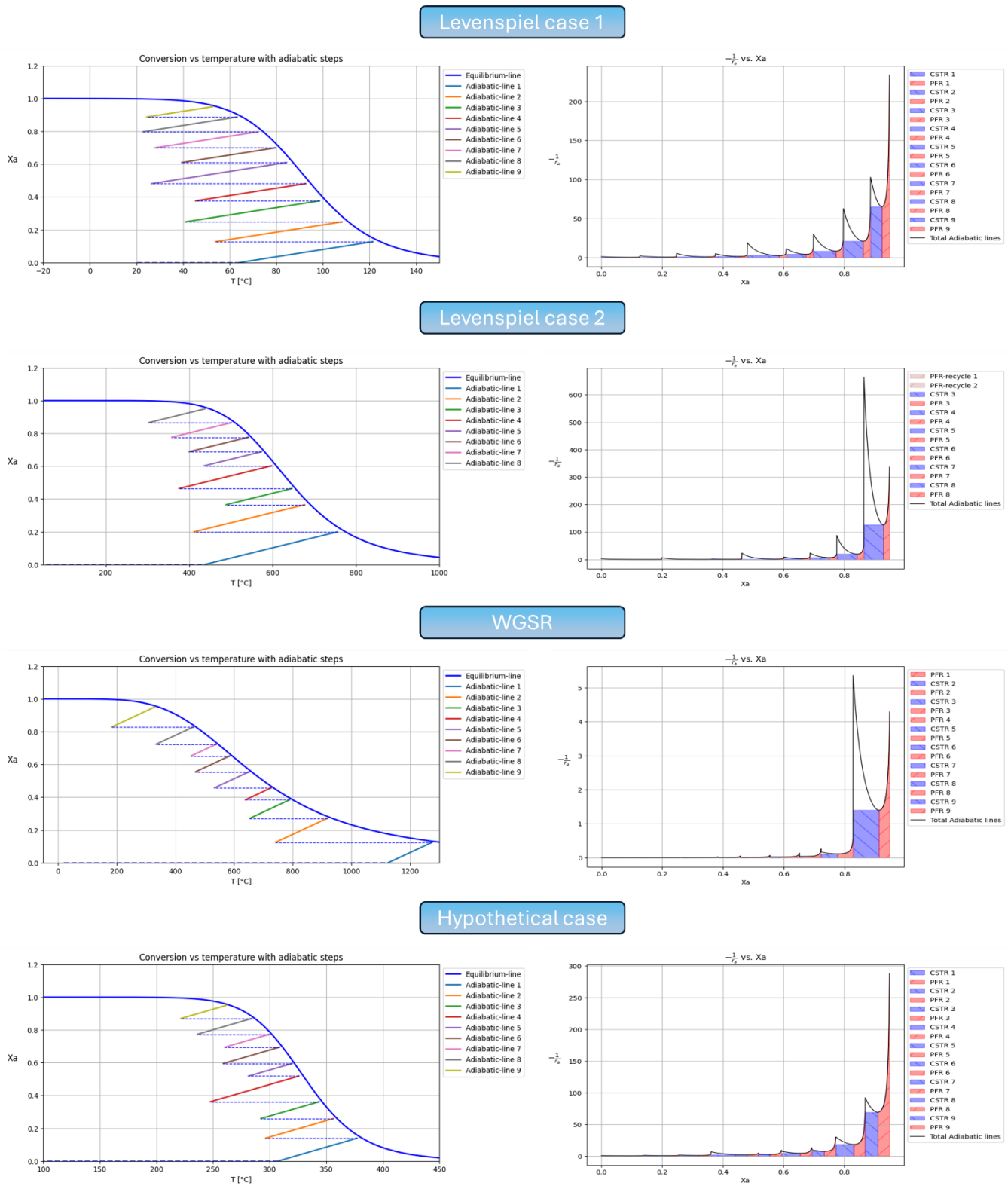


Figure 23: Optimal results based on a 10× scaling factor for surface area

Figure 24 illustrates the optimal configurations based on a surface area objective with 25 times the weight of the other objectives.

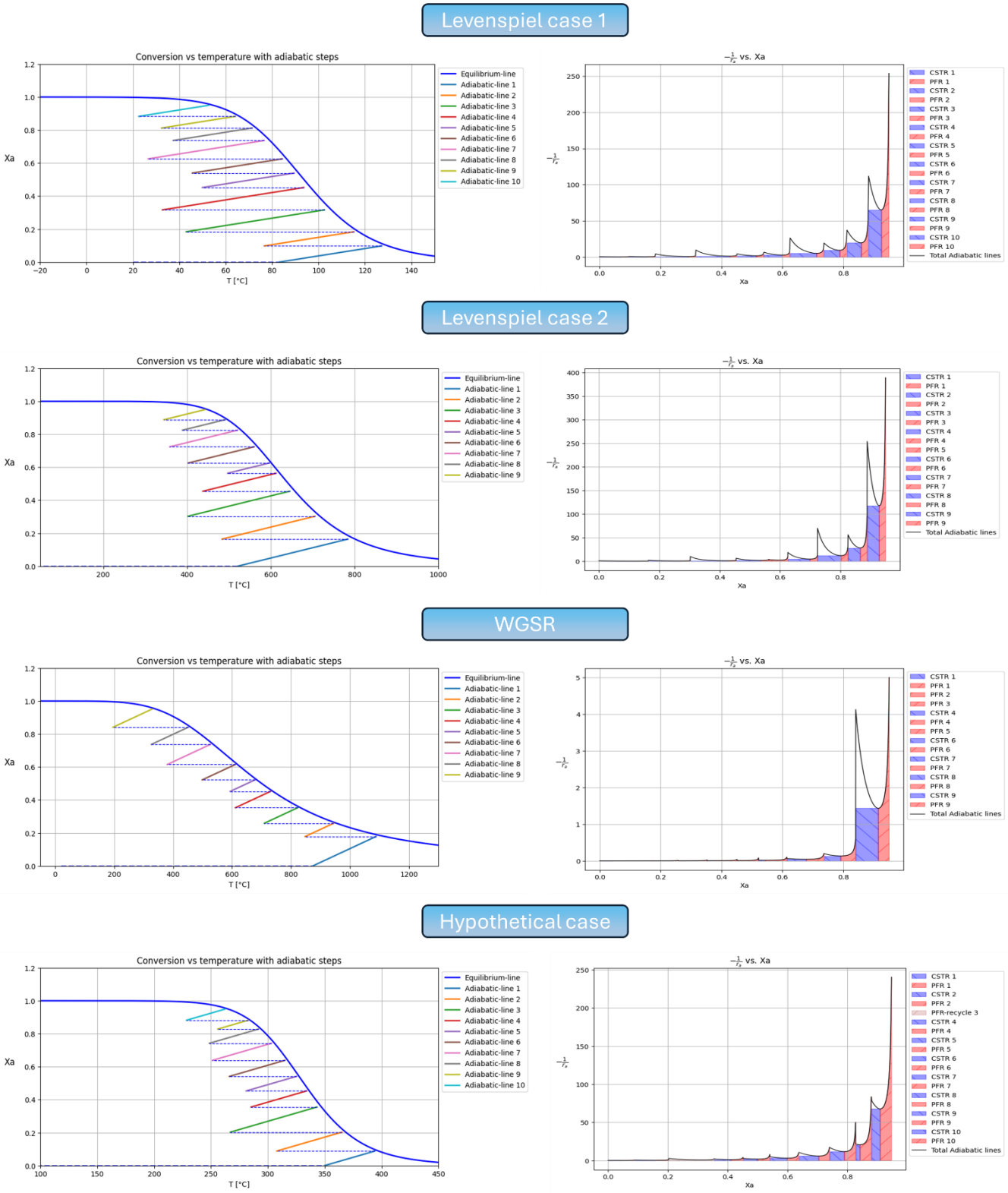


Figure 24: Optimal results based on a 25× scaling factor for surface area

Figure 25 shows the extreme case where the surface area objective is scaled 50 times higher than the others.

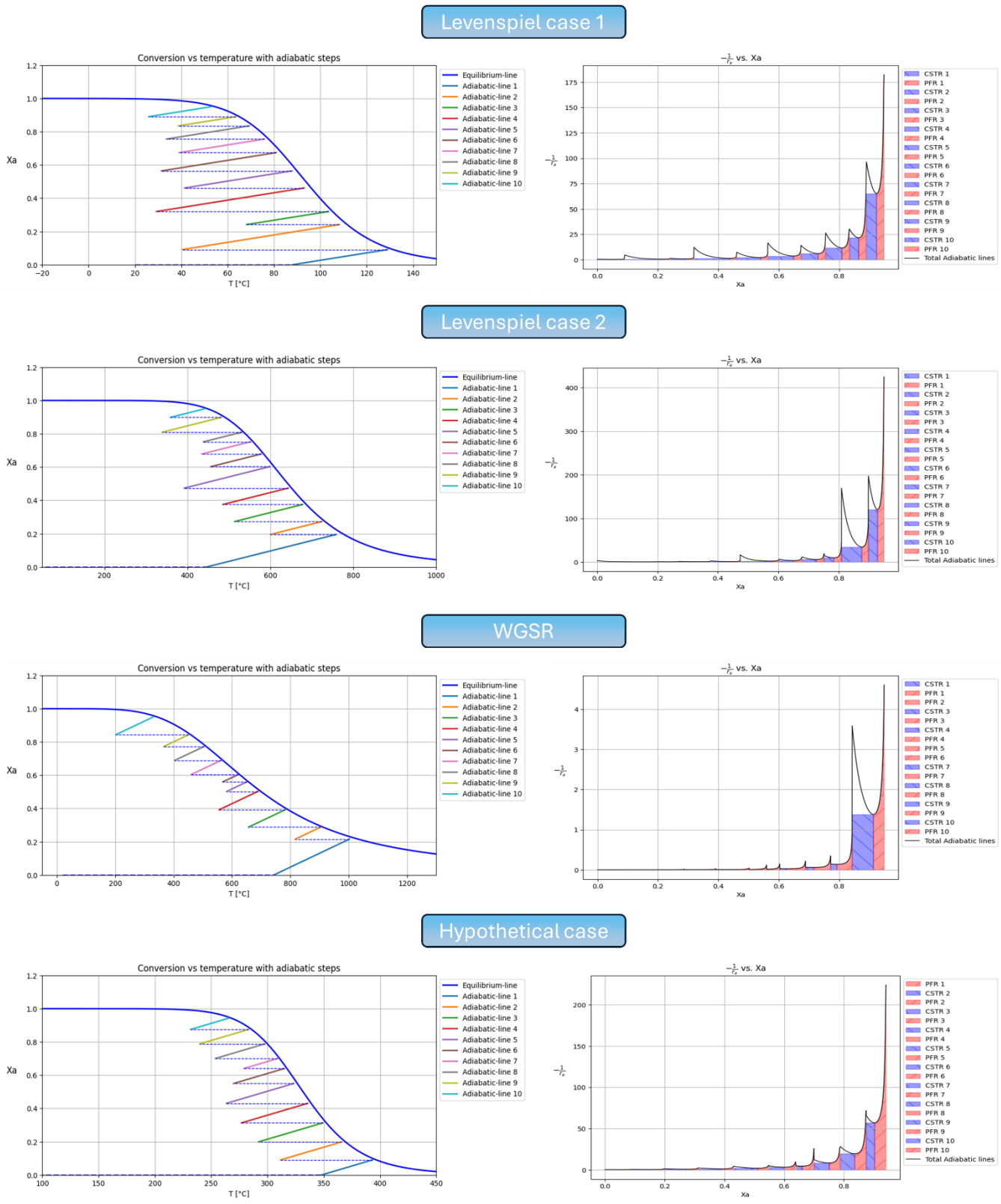


Figure 25: Optimal results based on a 50× scaling factor for surface area

Figure 26 shows the effect of the multiplication factor for the surface area objective on the optimal reactor configurations for each case. The surface area values for Case 3 were multiplied by 100 to bring them into the same range as those of the other cases, and to improve the readability of Figure 26a. In Figure 26b, the values for heat exchange were normalized based on the values at the initial scaling factors described in Section 4.1.

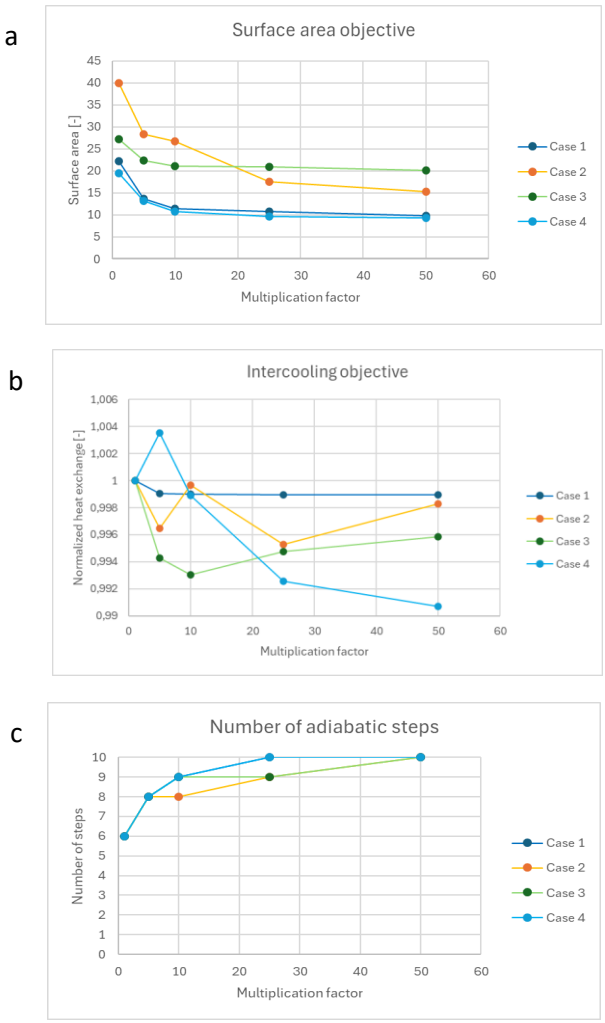


Figure 26: Influence of surface area objective weight on the optimal reactor configuration (a = surface area objective, b = intercooling objective, c = number of adiabatic steps objective)

As shown in Figures 21 to 26, increasing the weight of the surface area objective leads to optimal configurations with a greater number of adiabatic steps. When the multiplication factor is raised to five, the number of steps increases from six to eight for all case studies. This suggests that the algorithm reduces reactor volume by dividing the conversion into smaller increments, thereby enabling improved reaction rates.

As the scaling factor increases further to ten, twenty-five, and fifty times the initial value, the trend continues. Cases one and four reach the maximum of ten steps at a scaling factor of twenty-five, while cases two and three stabilize at nine steps at this scaling factor. This may be due to differences in reaction kinetics that limit the benefit of an additional step. When the surface area weight is increased by a factor of fifty, all cases converge to a configuration with ten adiabatic steps (Figure 26c). This shows that a higher surface area weighting results in more segmented configurations and a steady decrease in total surface area (Figure 26a), indicating effective reactor volume minimization by the algorithm. Interestingly, despite the increasing number of adiabatic steps, the total amount of

intercooling required remains nearly constant across all cases (Figure 26b). This suggests that the algorithm distributes the cooling load more evenly across the added steps, rather than increasing the overall energy demand for cooling.

4.3 Optimization preference toward intercooling requirements

To investigate the influence of the intercooling requirement objective on the resulting reactor configuration, the weight assigned to the fitness value for intercooling was systematically increased while keeping the weights of the surface area and step-count objectives constant. As with the previous analysis, the scaling factor was increased by multiplication factors of 5, 10, 25, and 50 compared to the baseline values used in Section 4.1.

Figure 27 illustrates the optimal reactor configurations for each case when the intercooling objective is given 5 times the weight of the other objectives.

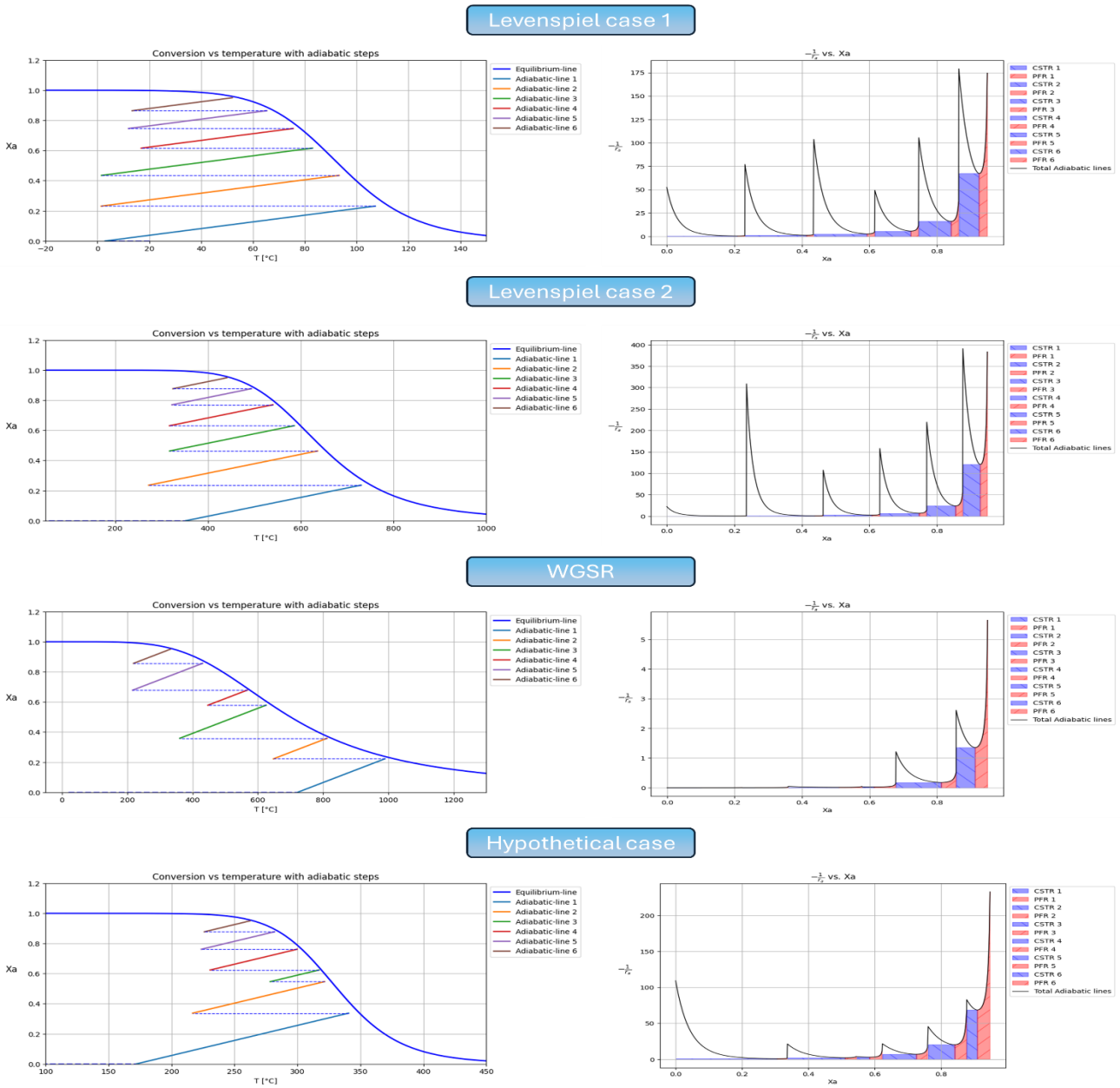


Figure 27: Optimal results based on a 5× scaling factor for intercooling

Figure 28 presents the optimal configurations for all cases when the intercooling objective weight is increased 10 times the weight of the other objectives.

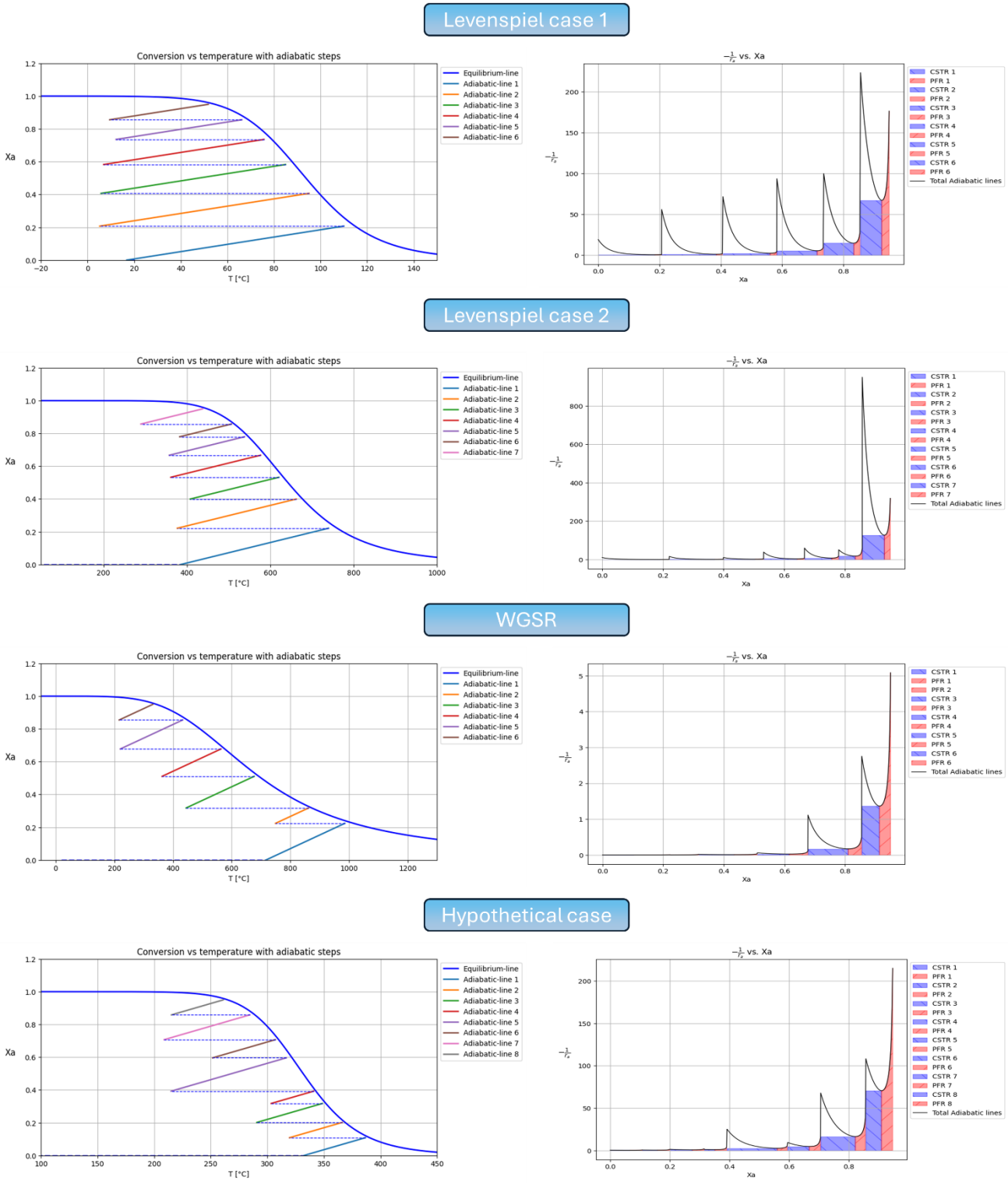


Figure 28: Optimal results based on a 10× scaling factor for intercooling

Figure 29 shows the optimal configurations when the intercooling objective is given 25 times the weight of the other objectives.

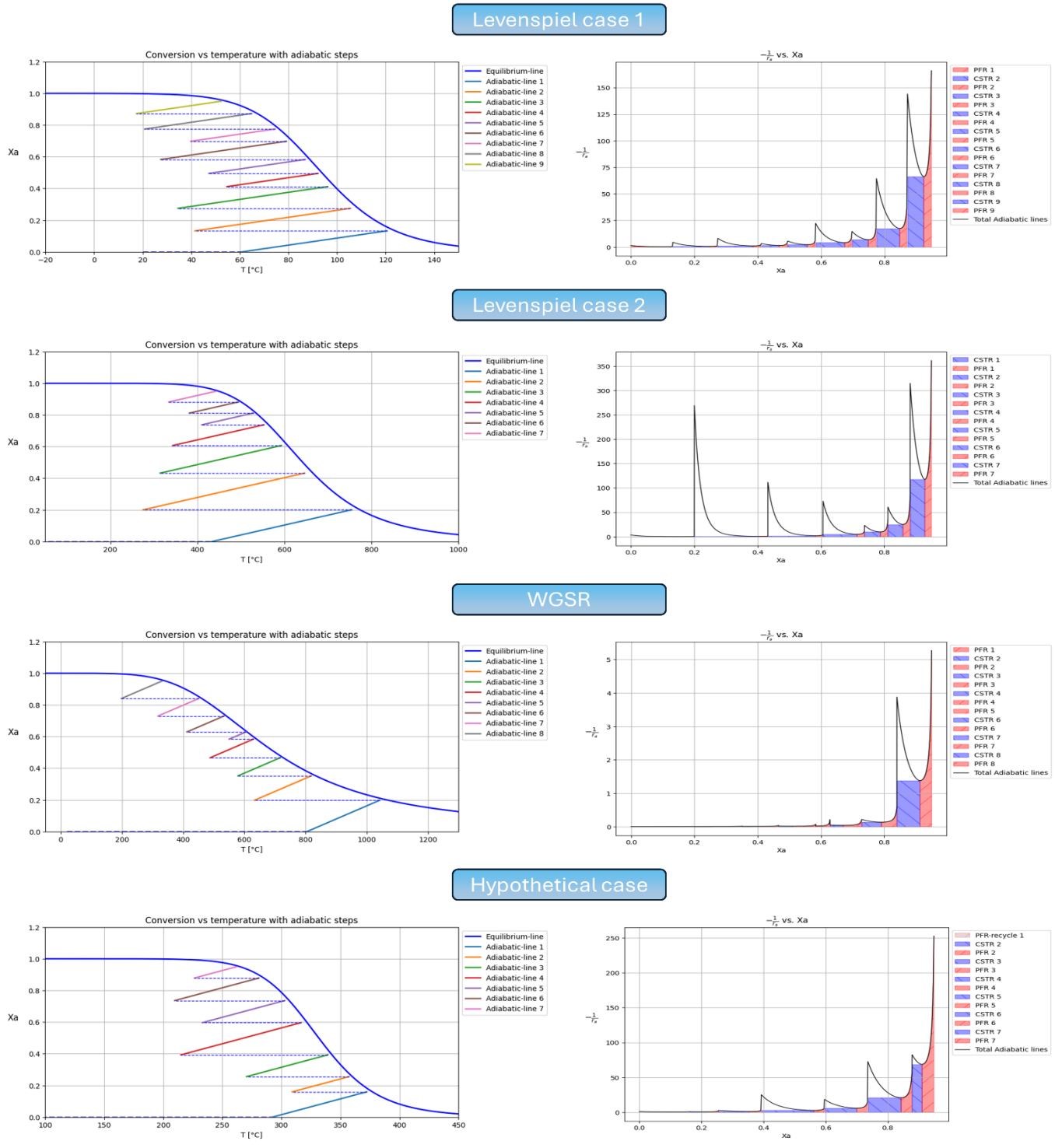


Figure 29: Optimal results based on a 25× scaling factor for intercooling

Figure 30 illustrates the optimal configuration for each case when the intercooling objective is increased 50 times the weight of the other objectives.

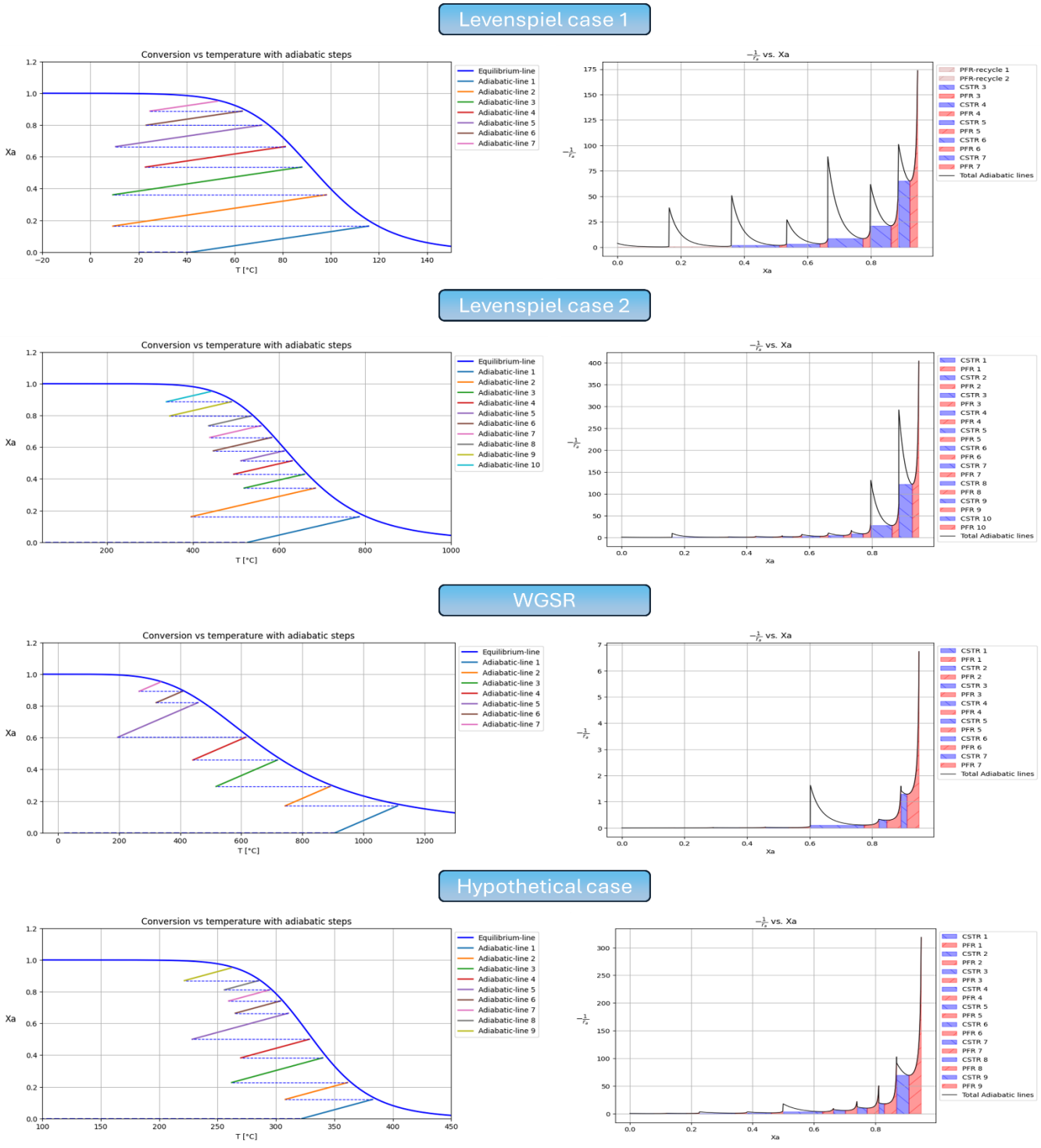


Figure 30: Optimal results based on a 50× scaling factor for intercooling

The influence of the objective weight for the intercooling demand on the optimal reactor configuration is illustrated in Figure 31 below. In this Figure, the same scaling and normalization were applied as in Figures 26a and 26b.

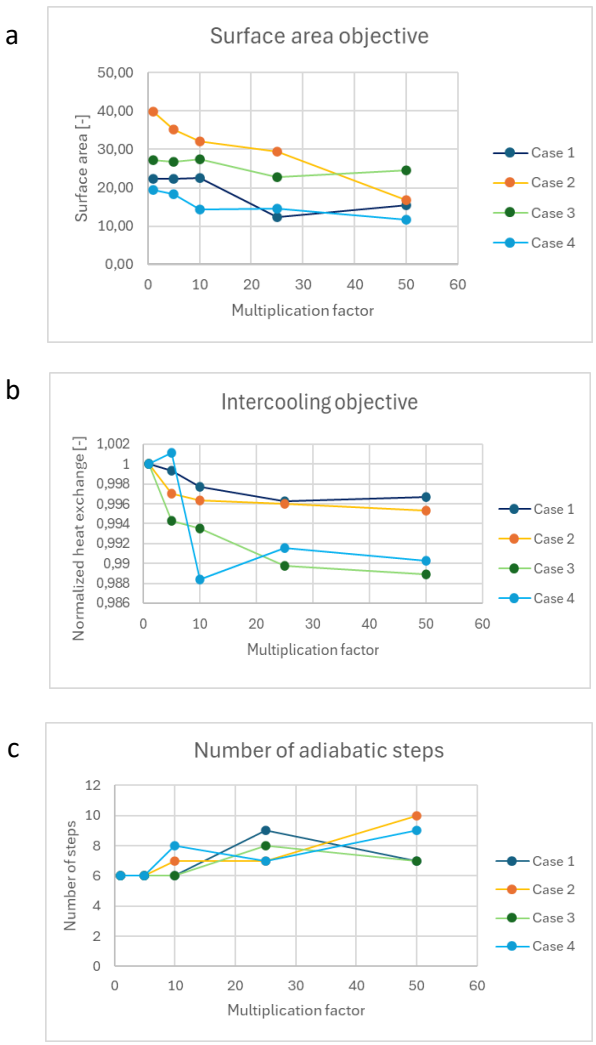


Figure 31: Influence of the multiplication factor for the intercooling objectives on the optimal configuration (a = surface area objective, b = intercooling objective, c = number of adiabatic steps objective)

The results in Figures 27 to 31 show that increasing the weighting of the intercooling objective slightly affects the overall structure of the optimal reactor configurations. While the number of adiabatic steps does not follow a consistent trend across all scaling factors, a slight tendency toward higher step counts can be observed at elevated scaling factor values. This suggests that the algorithm may favour more segmented configurations when intercooling minimization becomes increasingly dominant. One possible explanation is that dividing the conversion over more adiabatic steps allows each individual step to achieve a lower conversion increment. This, in turn, permits the first step to start at a higher inlet temperature and begin closer to the equilibrium line. As a result, the total cooling demand can be slightly lowered when the number of steps increases.

It is notable that the total required intercooling remains nearly constant across all scaling factors as shown in Figure 31.b. This outcome can be attributed to the modelling assumptions, where the total heat released is directly linked to the fixed overall conversion target of 0.95 and a constant heat of reaction. Because the enthalpy change is assumed to be independent of temperature or conversion, reaching the target conversion always results in approximately the same total heat release. Dividing the reaction into more steps has limited influence on the overall amount of heat that must be removed.

As the importance of minimizing intercooling increases, the algorithm tends to select a higher starting temperature to reduce the initial cooling of the feed, thereby slightly lowering the total cooling load.

4.4 Optimization preference toward number of adiabatic steps

To evaluate the impact of minimizing the number of adiabatic steps on the final reactor configuration, the weight of the fitness value associated with the step-count objective was gradually increased, while the weights for surface area and intercooling remained constant. The scaling factor for the step-count objective was raised by multiplication factors of 5, 10, 25, and 50 relative to the baseline values used in Section 4.1. This analysis examines how a stronger preference for reactor designs with fewer units affects the resulting optimal configurations.

Figure 32 presents the optimal reactor configuration for each case when the number of adiabatic steps objective is given 5 times the weight of the other objectives.

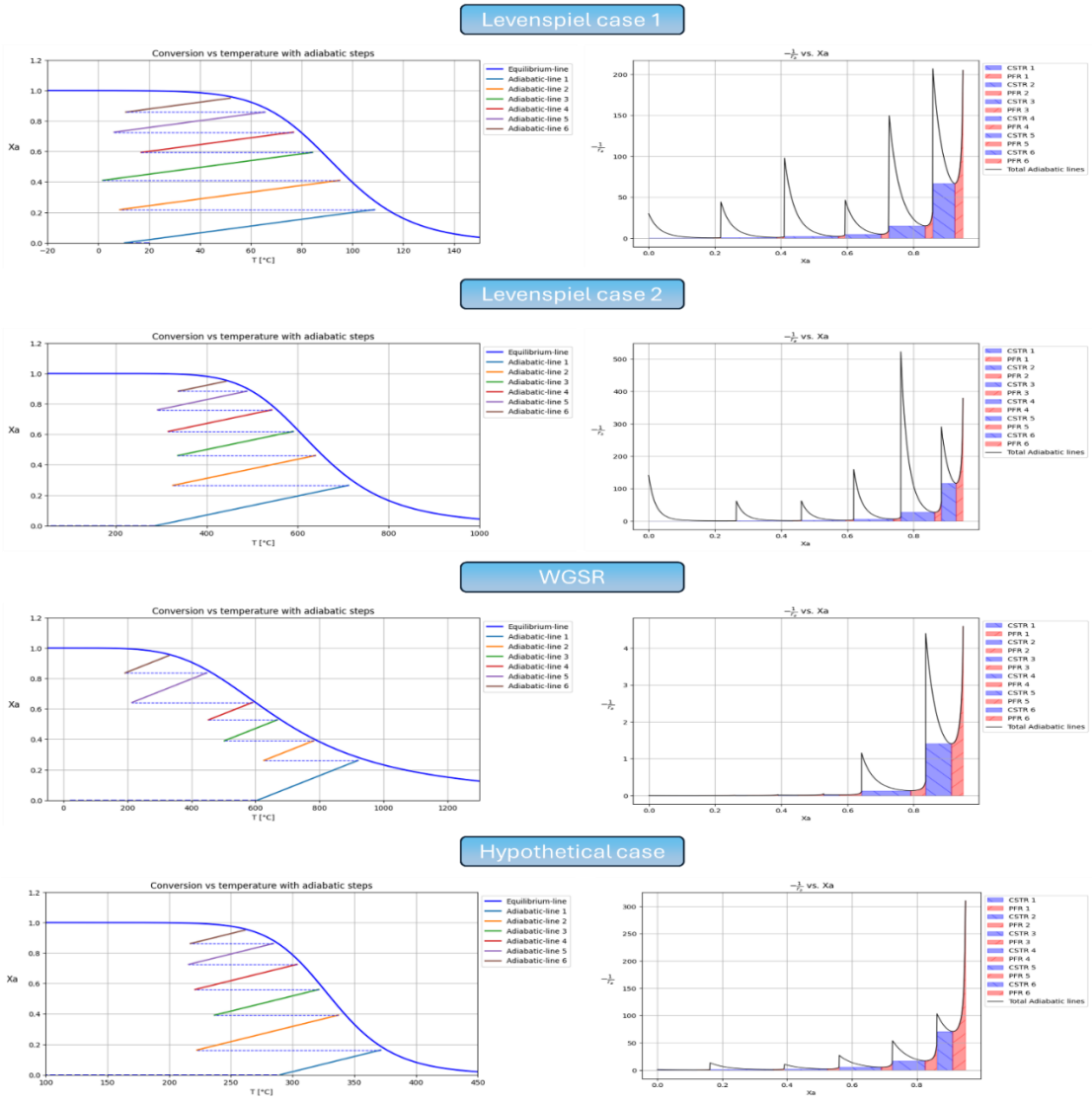


Figure 32: Optimal results based on a 5× scaling factor for number of steps

Figure 33 shows the optimal configurations when the number of adiabatic steps objective is given 10 times the weight of the other objectives.

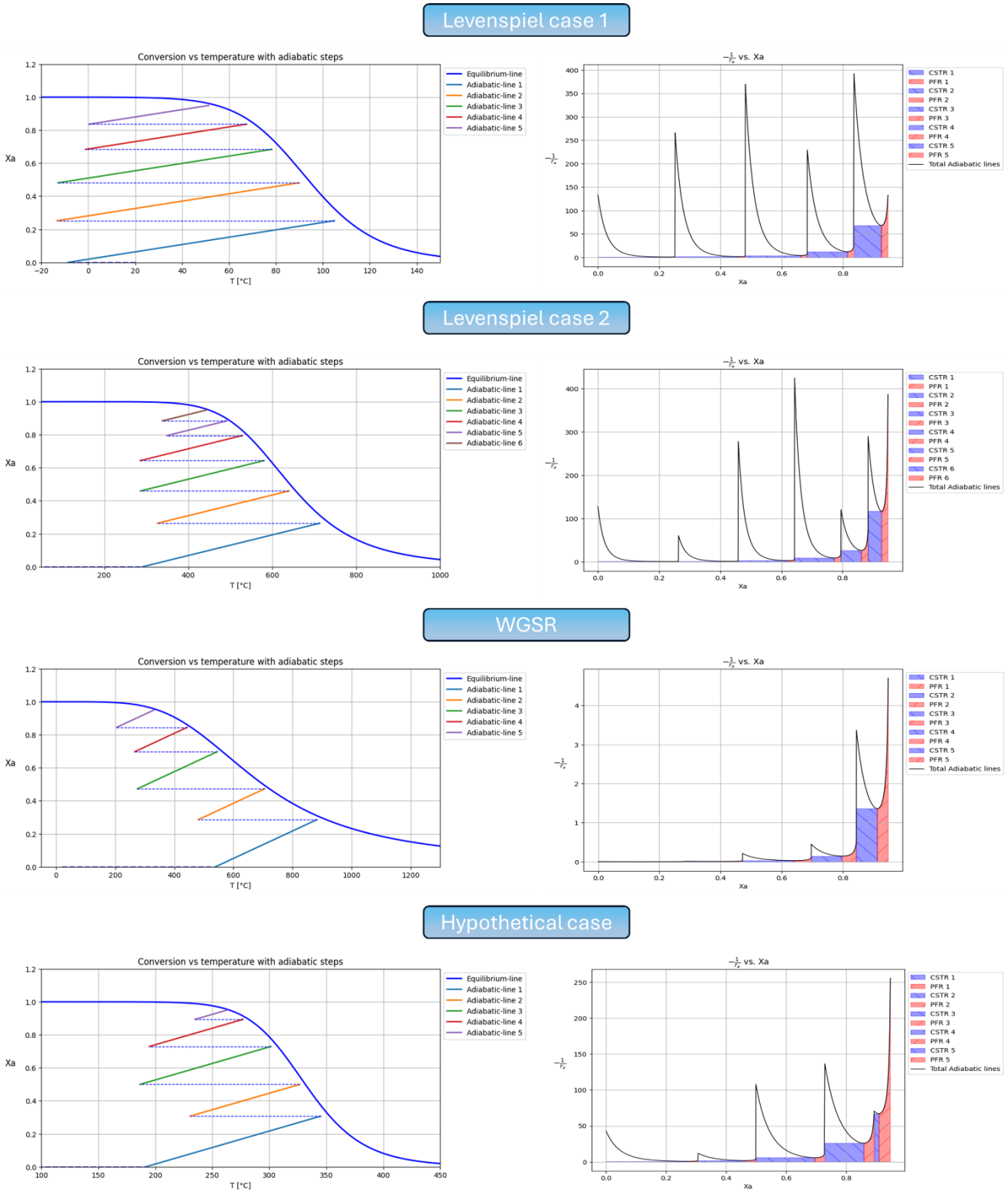


Figure 33: Optimal results based on a 10× scaling factor for number of steps

Figure 34 presents the optimal reactor configurations for each case when the number of adiabatic steps objective is given 25 times the weight of the other objectives.

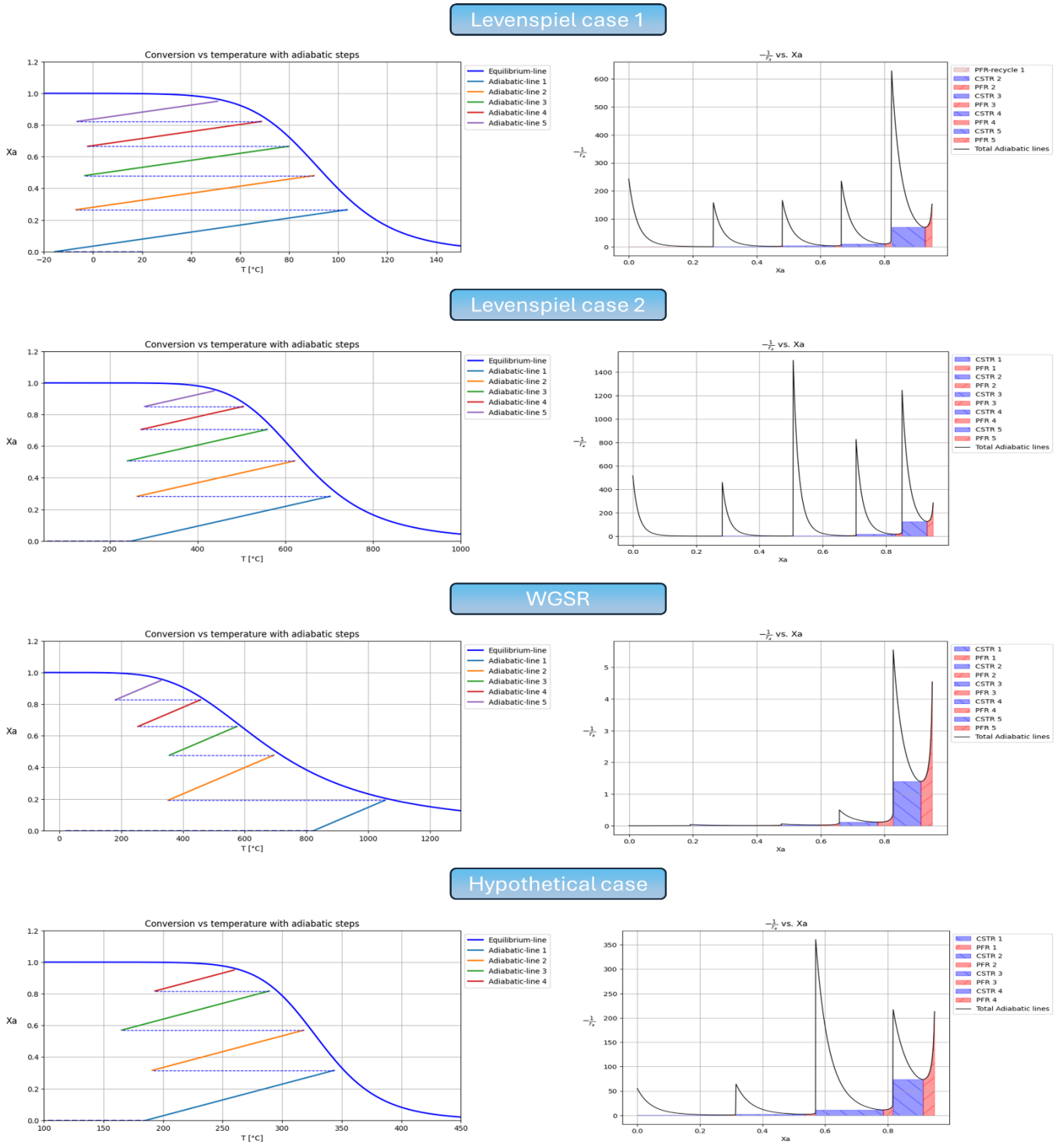


Figure 34: Optimal results based on a 25× scaling factor for number of steps

Figure 35 shows the optimal reactor configurations for each case when the number of adiabatic steps objective is given 50 times the weight of the other objectives.

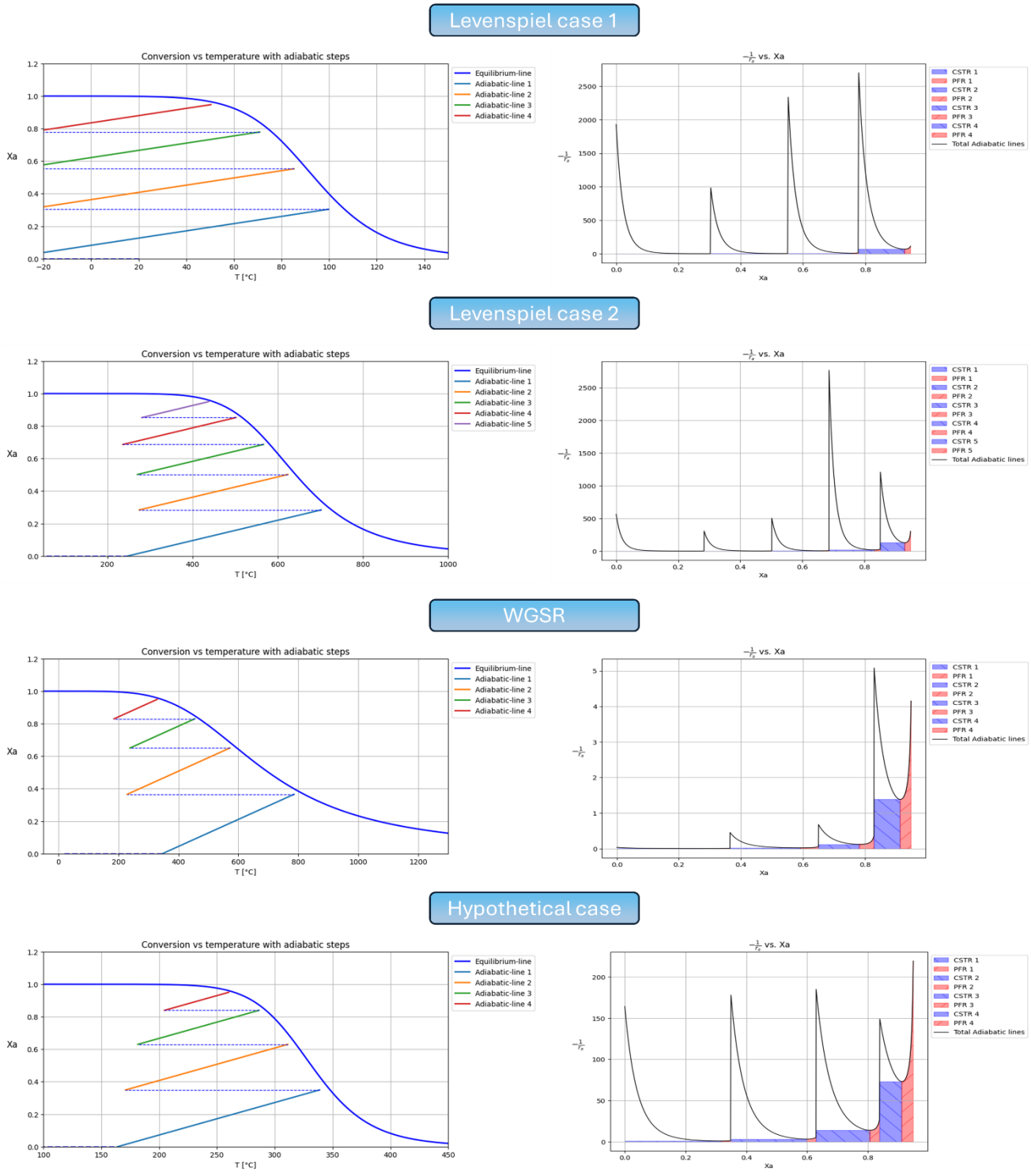


Figure 35: Optimal results based on a 50× scaling factor for number of steps

Figure 36 illustrates the influence of objective weighting for the number of adiabatic steps on the optimal reactor configuration for each case. In this Figure, the same scaling and normalization were applied as in Figures 26a and 26b.

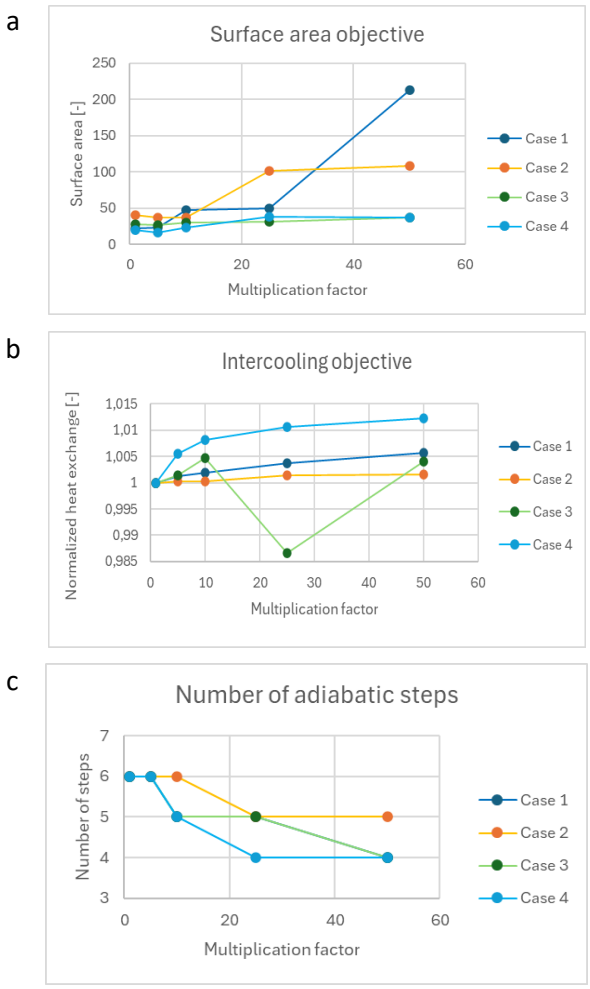


Figure 36: Effects of number of adiabatic steps objective weights on the optimal reactor configurations (a = surface area objective, b = intercooling objective, c = number of adiabatic steps objective)

Figures 32 and 36 illustrate that at a fivefold increase, the optimal configuration still consists of six adiabatic steps for all cases. This indicates that the algorithm continues to prioritize minimizing surface area and intercooling requirements, and that reducing the number of steps offers no overall benefit at this moderate weighting.

When the weight is increased tenfold as shown in Figures 33 and 36, the algorithm shifts to a five-step configuration for every case except Case 2. This suggests that the pressure to reduce the number of reactors has begun to influence the optimization, even if it results in slightly higher surface area. The new configuration reflects a different trade-off point, where fewer reactors are preferred despite some loss in efficiency.

As presented in Figures 34 and 36, further increasing the weight to 25 times still results in five steps for three cases, while Case 4 shifts to a four-step configuration. This indicates that for the first three cases, reducing the number of steps to four would lead to a noticeable drop in performance in terms of surface area or cooling requirements. In contrast, for Case 4, the trade-off becomes favourable enough to justify a more compact design. At this point, the algorithm starts to prioritize minimizing the number of steps more strongly, but only when the associated performance penalty remains acceptable.

At the highest weighting factor (fiftyfold in Figures 35 and 36), the algorithm returns a four-step configuration for three cases, while Case 2 still maintains five steps. This marks a clear shift in priority: minimizing the number of steps has become the dominant driver, even though it leads to larger reactors.

5. Conclusions

This thesis presents the development of a two-stage optimization algorithm based on standard genetic algorithms, aimed at automating the modelling and design of chemical reactor systems. The work addresses the challenge of optimizing complex reactor networks under multiple, conflicting objectives. Traditional reactor design methods rely heavily on manual tuning, which is time-consuming and may not effectively balance trade-offs between reactor volume, cooling requirements, and operational complexity.

The motivation behind this research is to provide a two-stage algorithm for multi-objective optimization that automatically identifies optimal reactor configurations aligned with specific process goals. The first stage employs a multi-objective genetic algorithm to determine the optimal number and sequence of adiabatic reaction steps needed to achieve a fixed target conversion of 0.95, balancing reactor volume, intercooling demands, and the number of steps to optimize efficiency and complexity. The second stage refines this configuration by selecting the optimal reactor types or combinations for each step using single-objective optimization, further minimizing reactor volume.

Compared to manual design approaches, the two-stage algorithm efficiently generates balanced reactor configurations in approximately 10 minutes per objective weight combination, dramatically reducing design time compared to traditional methods that can take an entire day and may still miss the true optimum.

The algorithm's performance was demonstrated through systematic variation of objective weights across multiple case studies, showing how shifting priorities influence the optimal configuration. Prioritizing reactor volume resulted in designs with more, smaller adiabatic steps. Focusing on intercooling emphasized optimizing initial step temperatures rather than step count. Emphasizing simplicity led to more compact configurations with fewer but larger steps.

These findings confirm that the proposed methodology effectively produces logically balanced reactor configurations tailored to user-defined priorities, offering process engineers a helpful tool for navigating complex design trade-offs. This work advances automated reactor design and lays the groundwork for future extensions to non-adiabatic conditions, endothermic reactions, parallel and series reactions, and the integration of economic objectives.

References

[1] O. Levenspiel and Department of Chemical Engineering, “Chemical Reaction Engineering”, John Wiley & Sons, 1999, New York, USA.

[2] A. K. Coker, “Non-Isothermal Reactors,” in *Modeling of Chemical Kinetics and Reactor Design*, Gulf Professional Publishing, 2001, pp. 424–551. doi: 10.1016/B978-088415481-5/50008-5

[3] C. M. Silva and E. C. Biscaia, “Genetic algorithm development for multi-objective optimization of batch free-radical polymerization reactors,” *Computers & Chemical Engineering*, vol. 27, no. 8–9, pp. 1329–1344, Apr. 2003, doi: 10.1016/s0098-1354(03)00056-5

[4] H. Kordabadi and A. Jahanmiri, “Optimization of methanol synthesis reactor using genetic algorithms,” *Chemical Engineering Journal*, vol. 108, no. 3, pp. 249–255, Apr. 2005, doi: 10.1016/j.cej.2005.02.023.

[5] A. V. Sobolev, A. S. Gazetdinov, and D. S. Samokhin, “Genetic algorithms for nuclear reactor fuel load and reload optimization problems,” *Nuclear Energy and Technology*, vol. 3, no. 3, pp. 231–235, Aug. 2017, doi: 10.1016/j.nucet.2017.07.002.

[6] A. D. Nandasana, A. K. Ray, and S. K. Gupta, “Applications of the Non-Dominated Sorting Genetic Algorithm (NSGA) in chemical reaction engineering,” *International Journal of Chemical Reactor Engineering*, vol. 1, no. 1, Apr. 2003, doi: 10.2202/1542-6580.1018.

[7] S. Verma, M. Pant, and V. Snasel, “A Comprehensive Review on NSGA-II for Multi-Objective Combinatorial Optimization Problems,” *IEEE Access*, vol. 9, pp. 57757–57791, Jan. 2021, doi: 10.1109/access.2021.3070634.

[8] T. Rodgers, “Chemical Reaction Engineering,” 2013. [Online]. Available: https://personalpages.manchester.ac.uk/staff/tom.rodgers/documents/CRE_Notes.pdf, accessed: [March 2025]

[9] Libretexts, “Reversible vs. Irreversible Reactions,” *Chemistry LibreTexts*, Jan. 30, 2023. https://chem.libretexts.org/Bookshelves/Physical_and_Theoretical_Chemistry_Textbook_Maps/Supplemental_Modules_(Physical_and_Theoretical_Chemistry)/Equilibria/Dynamic_Equilibria/Reversible_vs._Irreversible_Reactions, accessed: [March 2025].

[10] E. Baraj, K. Ciahotný, and T. Hlinčík, “The water gas shift reaction: Catalysts and reaction mechanism,” *Fuel*, vol. 288, p. 119817, Dec. 2020, doi: 10.1016/j.fuel.2020.119817.

[11] J. H. Lee and D. L. Trimm, “Catalytic combustion of methane,” *Fuel Processing Technology*, vol. 42, no. 2–3, pp. 339–359, Apr. 1995, doi: 10.1016/0378-3820(94)00091-7.

[12] Libretexts, “Limiting reagents,” *Chemistry LibreTexts*, Jun. 30, 2023. https://chem.libretexts.org/Bookshelves/Inorganic_Chemistry/Supplemental_Modules_and_Websites_(Inorganic_Chemistry)/Chemical_Reactions/Limiting_Reagents, accessed: [March 2025].

[13] Libretexts, “Le Chatelier’s Principle Fundamentals,” *Chemistry LibreTexts*, Jan. 30, 2023. https://chem.libretexts.org/Bookshelves/Physical_and_Theoretical_Chemistry_Textbook_Maps/Supplemental_Modules_(Physical_and_Theoretical_Chemistry)/Equilibria/Le_Chateliers_Principle/Le_Chatelier's_Principle_Fundamentals, accessed: [March 2025].

[14] "A guide to rate laws and reaction rates," *Applied Photophysics*. https://www.photophysics.com/knowledge-hub/a-guide-to-rate-laws-and-reaction-rates/, accessed: [March 2025].

[15] S. Liu, "An overview of chemical reaction analysis," in *Bioprocess Engineering*, 2016, pp. 81–137. doi: 10.1016/b978-0-444-63783-3.00003-4.

[16] "Plug Flow Reactor (PFR): Principle,Features,and Applications." https://www.blikai.com/blog/components-parts/plug-flow-reactor-pfr-principle-features-and-applications, accessed: [March 2025].

[17] Libretexts, "1.24: Plug flow reactors and comparison to continuously stirred tank reactors," *Chemistry LibreTexts*, Sep. 24, 2022. https://chem.libretexts.org/Courses/New_York_University/CHEM-UA_652%3A_Thermodynamics_and_Kinetics/01%3A_Lectures/1.24%3A_Plug_flow_reactors_and_comparison_to_continuously_stirred_tank_reactors, accessed: [March 2025].

[18] F. Filtration, "Chemical Reactors: The Ultimate FAQ Guide," Nov. 09, 2020. https://www.filsonfilters.com/ultimate-guide-of-chemical-reactors, accessed: [March 2025].

[19] "Continuous stirred tank reactors (CSTR) | Intro to Chemical Engineering Class Notes | Fiveable," *Fiveable*. https://fiveable.me/introduction-chemical-engineering/unit-8/continuous-stirred-tank-reactors-cstr/study-guide/rIuA3eugKFJcOcbe, accessed: [April 2025].

[20] Wikipedia contributors, "Continuous stirred-tank reactor," Wikipedia, Aug. 06, 2024. https://en.wikipedia.org/wiki/Continuous_stirred-tank_reactor, accessed: [April 2025].

[21] A. Zaidi, "Batch and Levenspiel plots for parallel and series reactors," *Engineeringness*, Jan. 19, 2025. https://engineeringness.com/batch-and-levenspiel-plots-for-parallel-and-series-reactors/, accessed: [April 2025].

[22] Libretexts, "2.5: reaction rate," *Chemistry LibreTexts*, Feb. 13, 2023. https://chem.libretexts.org/Bookshelves/Physical_and_Theoretical_Chemistry_Textbook_Maps/Supplemental_Modules_(Physical_and_Theoretical_Chemistry)/Kinetics/02%3A_Reaction_Rates/2.05%3A_Reaction_Rate, accessed: [April 2025].

[23] S. Li, "Tubular reactor," in *Reaction Engineering*. Elsevier, 2017, pp. 161–212. doi: 10.1016/B978-0-12-410416-7.00004-5

[24] S. Li, "Tank reactor," in *Reaction Engineering*, Elsevier, 2017, pp. 95–160. doi: 10.1016/B978-0-12-410416-7.00003-3

[25] A. Sircar, K. Yadav, K. Rayavarapu, N. Bist, and H. Oza, "Application of machine learning and artificial intelligence in oil and gas industry," *Petroleum Research*, vol. 6, no. 4, pp. 379–391, Jun. 2021, doi: 10.1016/j.ptlrs.2021.05.009.

[26] S. Khanmohammadi, O. Kizilkan, and F. Musharavati, "Multiobjective optimization of a geothermal power plant," in *Thermodynamic Analysis and Optimization of Geothermal Power Plants*, C. O. Colpan, O. Kizilkan, and F. Kucukgencay, Eds. Academic Press, 2021, pp. 279–291. doi: 10.1016/B978-0-12-821037-6.00011-1

[27] A. O. Lyakhov, A. R. Oganov, H. T. Stokes, and Q. Zhu, “New developments in evolutionary structure prediction algorithm USPEX,” *Computer Physics Communications*, vol. 184, no. 4, pp. 1172–1182, Dec. 2012, doi: 10.1016/j.cpc.2012.12.009.

[28] V. Kanade, “Genetic Algorithms - meaning, working, and applications,” *Spiceworks Inc*, Sep. 06, 2023. https://www.spiceworks.com/tech/artificial-intelligence/articles/what-are-genetic-algorithms/, accessed: [April 2025].

[29] E. P. D. S. Amorim, C. R. Xavier, R. S. Campos, and R. W. D. Santos, “Comparison between Genetic Algorithms and Differential Evolution for Solving the History Matching Problem,” in *Lecture notes in computer science*, 2012, pp. 635–648. doi: 10.1007/978-3-642-31125-3_48.

[30] S. Achiche, L. Baron, and M. Balazinski, “Scheduling exploration/exploitation levels in genetically-generated fuzzy knowledge bases,” *IEEE Annual Meeting of the Fuzzy Information, 2004. Processing NAFIPS ’04.*, pp. 401-406 Vol.1, Jan. 2004, doi: 10.1109/nafips.2004.1336316.

[31] E.-S. M. El-Kenawy, M. M. Eid, M. Saber, and A. Ibrahim, “MBGWO-SFS: Modified Binary Grey Wolf Optimizer based on Stochastic Fractal Search for feature selection,” IEEE Access, vol. 8, pp. 107635–107649, Jan. 2020, doi: 10.1109/access.2020.3001151

[32] S. Hou and H. Wen, “Modeling and solving of uncertain process abnormity diagnosis problem,” *Energies*, vol. 12, no. 8, p. 1580, Apr. 2019, doi: 10.3390/en12081580.

[33] M. Bianchetti and J. Marenco, “A branch-and-cut algorithm for the routing and spectrum allocation problem,” *Discrete Applied Mathematics*, vol. 339, pp. 107–126, Jun. 2023, doi: 10.1016/j.dam.2023.06.015.

[34] D. Grygar and R. Fabricius, “An efficient adjustment of genetic algorithm for Pareto front determination,” *Transportation Research Procedia*, vol. 40, pp. 1335–1342, Jan. 2019, doi: 10.1016/j.trpro.2019.07.185.

[35] J. M. L, R. A, J. R, D. K, and S. B. M, “Application of Multi Objective Genetic Algorithm for Optimization of Core Configuration Design of a Fast Breeder Reactor,” *Computer Reviews Journal*, vol. 3, pp. 170–187, Apr. 2019, [Online]. Available: http://eprints.nias.res.in/1950/ , accessed: [April 2025].

[36] B. Min, K. Kannan, and S. Srinivasan, “Quick screening of Pareto-Optimal operating conditions for expanding Solvent–Steam assisted gravity drainage using hybrid Multi-Objective Optimization approach,” *Energies*, vol. 10, no. 7, p. 966, Jul. 2017, doi: 10.3390/en10070966

[37] GeeksforGeeks, “NonDominated Sorting Genetic Algorithm 2 (NSGAII),” *GeeksforGeeks*, Jul. 29, 2024. https://www.geeksforgeeks.org/non-dominated-sorting-genetic-algorithm-2-nsga-ii/, accessed: [April 2025].

[38] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, “A fast and elitist multiobjective genetic algorithm: NSGA-II,” *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, Apr. 2002, doi: 10.1109/4235.996017

[39] R. Saravanan, S. Ramabalan, and C. Balamurugan, “Evolutionary optimal trajectory planning for industrial robot with payload constraints,” *The International Journal of Advanced Manufacturing Technology*, vol. 38, no. 11–12, pp. 1213–1226, Sep. 2007, doi: 10.1007/s00170-007-1169-7

[40] Unknown, “Standard Thermodynamic values.” [Online]. Available: https://www.drjez.com/uco/ChemTools/Standard%20Thermodynamic%20Values.pdf, accessed: [April 2025].

[41] Y. Choi and H. G. Stenger, “Water gas shift reaction kinetics and reactor modeling for fuel cell grade hydrogen,” *Journal of Power Sources*, vol. 124, no. 2, pp. 432–439, Jul. 2003, doi: 10.1016/s0378-7753(03)00614-1.

[42] “Non-dominated Sorting Genetic Algorithm III (NSGA-III) — DEAP 1.4.1 documentation.” https://deap.readthedocs.io/en/latest/examples/nsga3.html, accessed: [April 2025].

[43] OpenAI, "ChatGPT", GPT-4. [Online]. Available: https://chat.openai.com, accessed: [April 2025].

[44] Grammarly, "Grammarly", vers. 2025. [Online]. Available: https://www.grammarly.com, accessed: [April 2025].

List of Appendices

Appendix A: Reaction kinetics for the Levenspiel cases and hypothetical case

The reaction kinetics data for the two Levenspiel cases were obtained from the literature [1, pp. 218 and 437]. These cases are both elementary aqueous exothermic reactions described as A⇌R. The corresponding Arrhenius parameters for both cases are presented in Table 5. In addition to the Levenspiel cases and the WGSR, a hypothetical first order aqueous exothermic reaction was introduced as a fourth case to further test the performance of the two-step algorithm. For this synthetic case, higher activation energies were deliberately chosen to create a more challenging scenario. The Arrhenius parameters for this hypothetical reaction are also listed in Table 5.

Table 5: Arrhenius parameters of the two Levenspiel cases and hypothetical case

Cases	Arrhenius equation
Levenspiel case 1	
Forward rate constant	$k_1 = e^{17.34-\frac{48900}{R \cdot T}}$
Backward rate constant	$k_2 = e^{42.04-\frac{124200}{R \cdot T}}$
Levenspiel case 2	
Forward rate constant	$k_1 = e^{7-\frac{83700}{R \cdot T}}$
Backward rate constant	$k_2 = e^{18-\frac{167400}{R \cdot T}}$
Hypothetical case	
Forward rate constant	$k_1 = e^{17.73-\frac{83000}{R \cdot T}}$
Backward rate constant	$k_2 = e^{41.51-\frac{202600}{R \cdot T}}$