Department of Physics and Astronomy
Heidelberg University

Master Thesis in Physics
submitted by

Yannis Seemann

born in Hamm (Germany)

2022

Multidimensional parameter optimization in fluid dynamic simulations of heavy-ion collisions

This Master Thesis has been carried out by Yannis Seemann at the
Physikalisches Institut Heidelberg
under the supervision of
Prof. Dr. Silvia Masciocchi

Abstract

The evolution of the quark-gluon plasma (QGP) that forms in heavy-ion collisions has been shown to be described well by the theory of relativistic viscous fluid dynamics. Since the fluid dynamic properties of the QGP are not accessible experimentally in a direct way, they have to be inferred by modeling the evolution of the system. Such modeling based on event-by-event simulations of individual collisions has been demonstrated to reproduce the average yields of particles at midrapidity with specific transverse momentum (transverse-momentum spectra) observed in experiment, however, it is associated with a high computational cost. A newly developed software package for heavy-ion collisions called Fluidum provides modeling of the fluid dynamic evolution at low computational cost when given an initial entropy density profile as well as parameters describing the initial state and the evolution. An optimization of these parameters is performed, such that the produced model output fits best transverse-momentum spectra of pions, kaons, and protons from Pb-Pb collisions at $\sqrt{s_{NN}}$ =2.76 TeV at the LHC measured by ALICE. This is done within a bayesian framework inferring bayesian estimates and uncertainties for each of the parameters using a feed-forward neural network ensemble emulator model and Markov chain Monte Carlo simulations, fully developed in the context of this thesis.

Zusammenfassung

Die Dynamik des Quark-Gluon-Plasmas (QGP), das sich bei Schwerionenkollisionen bildet, lässt sich nachweislich gut durch die Theorie der relativistischen viskosen Fluiddynamik beschreiben. Da die fluiddynamischen Eigenschaften des QGP experimentell nicht direkt zugänglich sind, müssen sie durch Modellierung der Evolution des Systems abgeleitet werden. Es hat sich gezeigt, dass eine solche Modellierung, die auf Event-für-Event Simulationen einzelner Kollisionen beruht, die im Experiment beobachteten durchschnittliche Teilchenanzahlen bei mittlerer Rapidität mit spezifischem Transversalimpuls (auch Transversalimpulsspektren genannt) reproduzieren kann, allerdings häufig mit einem hohen Rechenaufwand verbunden ist. Ein neu entwickeltes, auf Fluiddynamik basierendes Softwareframework namens Fluidum ermöglicht die Modellierung der fluiddynamischen Evolution des Systems mit geringem Rechenaufwand. Dafür müssen lediglich ein initiales Entropiedichteprofil sowie Parameter, die den Anfangszustand und die Evolution beschreiben, definiert werden. In dieser Arbeit werden diese Parameter optimiert, sodass das theoretische Modell die Transversalimpulsspektren von Pionen, Kaonen und Protonen aus Blei-Blei-Kollisionen bei $\sqrt{s_{NN}}$ =2.76 TeV, gemessen durch das ALICE Experiment am LHC, bestmöglich reproduziert. Dafür werden innerhalb einer neu entwickelten Bayesianischen Analyse Bayesianische Schätzungen und Unsicherheiten der Parameter unter Verwendung eines Ensembleemulators aus neuronalen Netzen und einer Markovketten Monte Carlo Simulation abgeleitet.

Contents

Chapter 1

Introduction

High-energy heavy-ion collisions at the Large Hadron Collider (LHC) at the European Organization for Nuclear Research (CERN) provide the opportunity to study Quantum Chromodynamics (QCD), the quantum field theory (QFT) of the strong interaction, at high temperatures and energy densities. Under these conditions, a new state of matter is formed, the so-called quark-gluon plasma (QGP), in which the quarks and gluons are deconfined and move almost freely within the strongly coupled plasma. According to the generally accepted model of the origin of the universe, the matter in the universe existed in the state of QGP within the time from 10^{-12} to 10^{-5} seconds after the Big Bang [1]. Thus, understanding the formation and evolution of the QGP can provide important information about the formation and evolution of the universe. Experimentally, the QGP can be produced in the laboratory via heavy-ion collisions and is examined in large experimental programs at CERN and other facilities around the world. Specifically designed to study the physics of strongly interacting matter at large energy densities where the QGP forms, is ALICE (A Large Ion Collider Experiment), one of the four major experiments at CERN [1]. However, the QGP is inaccessible via direct measurements because it exists only for a short period of time and in a small region of space. Only the final state of the free-streaming particles can be captured by the detectors after the system has undergone an evolution with complicated dynamics. To access important characteristics of the QGP nonetheless, the evolution is modeled theoretically and the output can be compared to experimental observables.
It has already been demonstrated, that the evolution of the QGP can be described by relativistic fluid dynamics [2]–[4]. Based on this description, the characteristics of the QGP have been examined recently based on a Bayesian analysis [5], [6], which is however severely limited by the computational expense of the simulation model.
In this thesis, a Bayesian analysis will be performed with a much more efficient software package called Fluidum [7], which solves the fluid dynamic equations of motion based on a mode expansion approach. Together with the phenomenological model Trento [8], which is used to produce the initial conditions for Fluidum, and the model FastReso [9], which describes the hadronization of the fluid field and therefore the production of the final particles, a theoretical model is formed that expresses the whole evolution of a heavy-ion collision based on a few parameters. In this analysis, the goal is to optimize these parameters using ALICE data of Pb-Pb collisions at $\sqrt{s_{\mathrm{NN}}} = 2.76\,\mathrm{TeV}$. In this context, a Bayesian optimization procedure adapted to the more efficient simulation model is developed, which includes an ensemble of neural networks for emulation.
In pursuit of this goal, the thesis has the following structure: In chapter 2, the theoretical groundwork is laid by the introduction of the physics of the QGP and heavy-ion collisions. After that, in chapter 3, the modeling of heavy-ion collisions is elaborated on, including the initial conditions, the fluid dynamic evolution of the QGP, and the hadronization. In chapter 4, the machine learning methods that are used within this work are introduced. This comprises neural networks and Markov-chain Monte-Carlo simulations. After that, the analysis is developed and performed in chapter 5, before in the results are discussed in chapter 6 and a conclusion is given in chapter 7.

Chapter 2

The Standard Model and Quantum Chromodynamics

The Standard Model of particle physics (SM) is a theory classifying all elementary particles as well as covering three of the four fundamental forces of the universe. It includes the strong, the weak, and the electromagnetic force while the gravitational force is currently not described by it. The elementary particles are categorized into groups according to their characteristic masses and quantum numbers. The SM includes twelve spin-1/2 particles (fermions) and thirteen particles with integer spin (bosons). The fermions can be divided based on their interaction: leptons interact via the electromagnetic and weak force and quarks can additionally interact via the strong force. The different species of leptons and quarks are organized into three generations with two particles each. Quarks form composite particles called hadrons that are made of two or more quarks. Mesons refer to quark anti-quark pairs and baryons refer to composite particles with three quarks. Among the bosons in the SM, there are twelve gauge bosons of spin 1, which act as the force carriers of the interactions: the photon mediates the electromagnetic interaction between charged particles, the W^+, W^- and Z bosons mediate the weak interaction between particles of different flavors, and the eight gluons mediate the strong interaction between quarks. Furthermore, there is a special boson, the Higgs boson, which has a unique role in the SM because it gives some particles a mass via the Higgs mechanism. A collection of all particles covered by the SM and their characteristics is given in fig. 2.1.

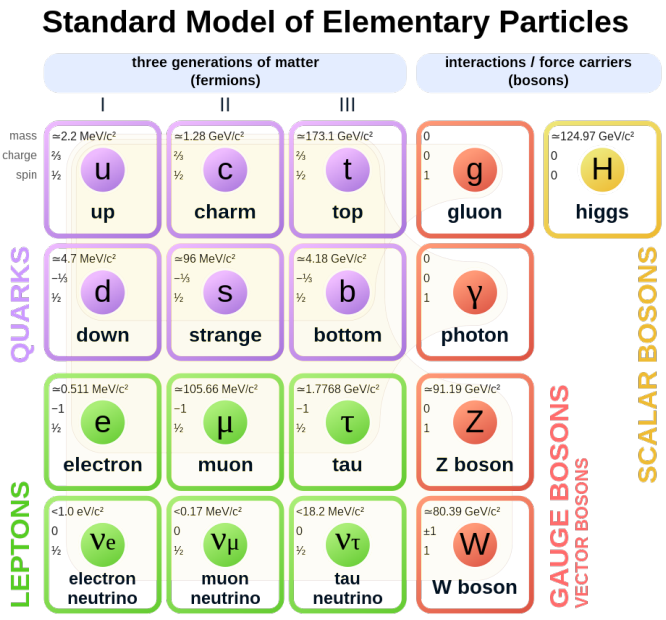


FIGURE 2.1: The elementary particles of the Standard Model of particle physics. Taken from [10].

Mathematically, the SM can be expressed in terms of a non-abelian gauge theory within the framework of quantum field theory (QFT). In quantum field theory, each particle is described as an excitation of a corresponding quantum field and all interactions are described by interaction terms in the Lagrangian density involving these quantum fields. As all field theories, also the SM is based on a certain set of symmetries. The symmetry that predominantly defines the SM is the local U(1)×SU(2)×SU(3) gauge symmetry, where the three factors are connected to the three interactions of the SM. The last term, the local SU(3) symmetry, relates to the strong interaction, and the part of the SM describing this is called Quantum Chromodynamics (QCD). Since QCD provides the framework to describe the formation of the quark-gluon plasma, which is the subject of this thesis, it will be laid out in more detail in the next section.

2.1 Quantum Chromodynamics

Quantum Chromodynamics is the quantum field theory of the strong interaction. The particles that are involved in the theory are the six quarks that can be classified by their masses and charges into three generations. The first generation contains the up (u) and down (d) quarks, the second the charm (c) and strange (s) quarks, and the third the top (t) and bottom (b) quarks. These species are also referred to by their flavors. In QCD, each flavor of quark comes in three different *colors*, which are introduced as the charges of the strong interaction. The color charge states are called red (r), blue (b), and green (g), nevertheless they are completely unrelated to the normal meaning of color and are rather a notation of the charge states. In QCD, the quarks may then be represented by the quark fields Ψ^f_C ($f = 1, 2, ..., 6$, $C = r, g, b$), which are the quantum fields for the flavor and color states. The color states of the quarks give rise to the $SU(3)_C$ symmetry of QCD, this can be expressed mathematically by the invariance of the color triplets $\Psi^f = (\Psi^f_r, \Psi^f_g, \Psi^f_b)^T$ under SU(3) local transformations [11]:

$$\Psi^f(x) \rightarrow \Psi'^f(x) = \exp[i\, g_s\, \alpha(x) \cdot \hat{T}]\Psi^f(x). \tag{2.1}$$

In this equation, $\alpha(x)$ are eight functions of the space-time coordinate x and $\hat{T} = \{T_a\}$ are the eight generators of the SU(3) symmetry, also related to the Gell-Mann matrices λ_a by

$$T_a = \frac{1}{2}\lambda_a, \quad a = 1, ..., 8. \tag{2.2}$$

g_s is the coupling strength, which can also be expressed in terms of the fine-structure constant of the strong interaction α_s by

$$\alpha_s = \frac{g_s^2}{4\pi}. \tag{2.3}$$

The eight generators of the $SU(3)_C$ symmetry can be associated to eight massless vector bosons, the gluons, which mediate the strong force between colored particles. In contrast to the force carrier of Quantum Electrodynamics (QED), the photon, gluons carry charge themselves, such that they can also participate in the strong interaction in addition to mediating it. They carry a pair of color and anti-color, leading in total to eight independent color combinations for the eight gluons. Since gluons carry color and mediate the strong force between colored particles, gluons can also interact with other gluons. This self-interaction in QCD has some important implications and makes QCD significantly harder to analyze than QED. In QCD, the gluons are associated to the gluon fields G^a_ν, where $a = 1, 2, ..., 8$ specifies the gluon color charge while $\nu = 1, 2, 3, 4$

denotes the space-time component. The dynamics of the gluons can be described by the gluon field strength tensor $G^a_{\mu\nu}$ using the gluon fields G^a_ν and the structure constants f_{abc}:

$$G^a_{\mu\nu} = \partial_\mu G^a_\nu - \partial_\nu G^a_\mu + g_s f_{abc} G^b_\mu G^c_\nu \tag{2.4}$$

The last term in eq. 2.4 accounts for the gluon self-interaction. It occurs in the theory since the generators of the SU(3) symmetry group do not commute ($[T^a, T^b] = f_{abc}T^c \neq 0$). The full dynamics of QCD are expressed in the Lagrangian density $\mathcal{L}_{\mathrm{QCD}}$ (for a given species of quarks), which is given by [12]

$$\mathcal{L}_{\mathrm{QCD}} = \bar{\Psi}\left(i\gamma^\mu D_\mu - m\right)\Psi - \frac{1}{4}G^a_{\mu\nu}G^{a,\mu\nu}, \tag{2.5}$$

where the first part accounts for the quark dynamics and the second part describes the gluon interactions. D_μ is the gauge covariant derivative which is given by

$$D_\mu = \partial_\mu - ig_s\frac{\lambda_a}{2}G^a_\mu \tag{2.6}$$

and couples the quark field with the coupling strength g_s via the generators T_a to the gluon fields. An important feature of non-abelian gauge theories as QCD that is connected to the gluon self-interaction is asymptotic freedom, in which the strength of the QCD coupling constant α_s varies with the momentum transfer q^2 of the interacting particles. Whereas for small momentum transfers the coupling strength is rather large, for large momentum transfers, respectively at large energy scales, the coupling strength decreases. For example, the coupling strength at $|q| \sim 1\,\mathrm{GeV}$ is $\mathcal{O}(1)$ and at $|q| > 100\,\mathrm{GeV}$ is $\mathcal{O}(0.1)$. Mathematically, the running of alpha can be treated by renormalization theory, which leads to the following behavior [11]:

$$\alpha_s(q^2) = \frac{\alpha_s(\mu^2)}{1 + \frac{33-2N_f}{12\pi}\alpha_s(\mu^2)\ln\left(\frac{q^2}{\mu^2}\right)}. \tag{2.7}$$

N_f is the number of quark flavors and μ^2 is an arbitrary reference scale. Since $N_f \leq 6$, $33 - 2N_f$ is always greater than zero and hence α_s decreases with increasing q^2 for QCD. This behavior has been verified experimentally as shown in fig. 2.2. Another consequence of the gluon self-interaction respectively the non-abelian nature of QCD is that for low momentum transfers or at low energy scales, the quarks and gluons are strongly bound within colorless singlet states. This is known as color confinement and is the reason why quarks cannot be observed in isolation but are always found in hadrons like mesons and baryons at temperatures below the Hagedorn temperature of $\sim 150\,\mathrm{MeV}$. This behavior changes for temperatures above the Hagedorn temperature, where the quarks and gluons are no longer confined. Then the system reaches a new type of matter which will be laid out in the next section.

Analytically, solutions to QCD processes are hard or impossible to obtain because of the features of the strong force. Therefore, perturbative and numerical methods are usually employed to solve QCD equations. For large momentum transfers, where the coupling constant is much smaller than one, perturbation theory can be applied. The equations may then be expanded in powers of the coupling constant and a finite number of the leading terms may be already sufficient to approximate the solution. This method is called perturbative QCD (pQCD) in the context of QCD. However, for small momentum transfers, the perturbative approach is no longer applicable since the coupling constant is $\mathcal{O}(1)$. In this regime, a computational technique, the so-called lattice

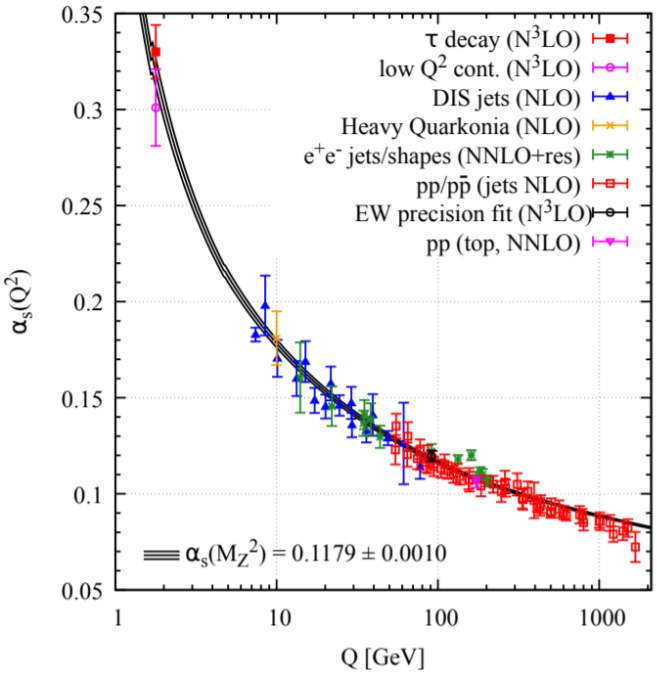


FIGURE 2.2: Summary of the measurements of α_s as a function of the energy scale Q. Taken from [14].

QCD (lQCD) has been established. In lQCD, the quantum-mechanical calculations are performed on a discrete lattice of space-time points. For an infinite number of lattice points and infinitesimal small distances between them the continuum QCD solution would be recovered. lQCD calculations are computationally intensive and have to be performed at supercomputing facilities, nevertheless, they have been successful in describing QCD in the non-perturbative regime. For example, the proton mass was calculated with a precision of less than two percent using lQCD [13]. However, lattice QCD is primarily applicable at low matter densities because at large densities the calculations interfere with a technical problem called the numerical sign problem. Therefore the strong interaction is only accessible by QCD calculations in limited ranges.

2.2 Quark-gluon Plasma

At large energy densities, the coupling strength of the strong interaction decreases due to the asymptotic freedom. The quarks and gluons of a system reaching this state become deconfined and are no longer bound to colorless hadrons. The system is then characterized by the dynamics of the color-charged quarks and gluons. Because of its similarity to plasma, this new state of matter is called quark-gluon plasma (QGP). The QGP phase can be reached for temperatures T at QCD scale ($\sim 200\,\mathrm{MeV}$) and/or if the matter density rises to the point where the hadrons overlap, such that the association of quarks to specific hadrons is meaningless. This happens if the inter-quark distances are below $\sim 1\,\mathrm{fm}$, which is approximately the size of a hadron. The density is often expressed using the baryon chemical potential μ_B, which quantifies the net baryon content of the system. Using the thermodynamic quantities of temperature and baryon chemical potential, the different phases of a strongly interacting matter can be illustrated in a phase diagram. Such a QCD phase diagram is depicted in fig. 2.3.
For low temperature and small baryon chemical potential, quarks and gluons are confined into color-neutral hadronic states. Normal nuclear matter is positioned within

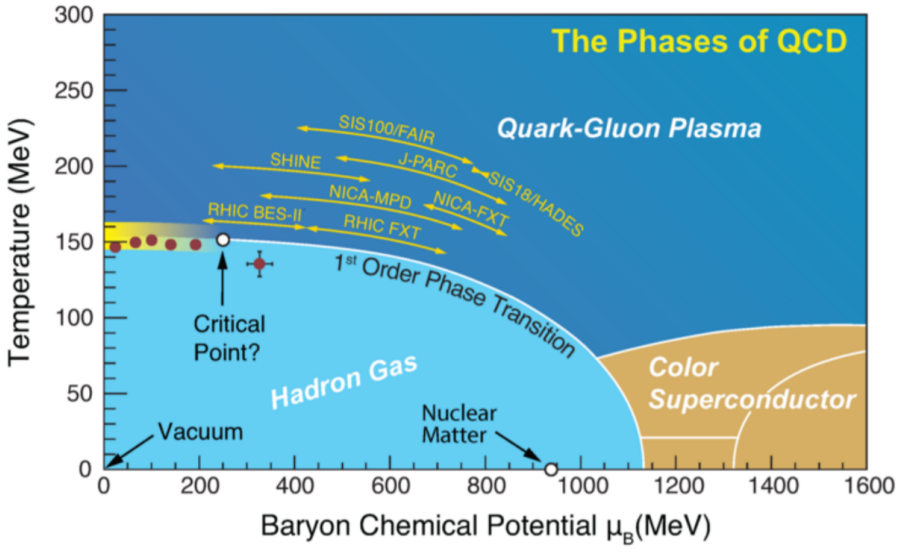


FIGURE 2.3: The QCD phase diagram covering the different phases of quark matter. Taken from [15].

this region at $T \sim 0$ and $\mu_B \sim 1\,\text{GeV}$. By increasing the temperature, at some point, the system reaches the state of the QGP. For vanishing baryon chemical potential μ_B, lattice QCD calculations have demonstrated that the phase transition between the hadronic and QGP state is a continuous crossover rather than a sharp phase transition [16]. Furthermore, the critical phase transition temperature where this crossover takes place, was estimated by lattice QCD calculation to be at $T_c = 156.5 \pm 1.5\,\text{MeV}$ [17]. The region at zero baryon chemical potential is of particular interest since the matter in the universe existed in the state of the QGP with $\mu_B \approx 0$ within the first few microseconds after the Big Bang according to the generally accepted model of the origin of the universe. Therefore, studying the QGP and its crossover to hadronic matter gives key insights into the formation of the universe. At larger baryon chemical potentials the experimental and theoretical understanding of the phase diagram is limited. Lattice QCD calculations are limited to the region where $\mu_B < T$ because of the numerical sign problem and perturbative QCD is not applicable for low temperatures. Nevertheless, at larger μ_B the crossover is expected to change into a first-order transition passing a critical second-order point where the continuous crossover turns into a sharp phase transition [16]. For temperatures below $\sim 100\,\text{MeV}$ and baryon chemical potentials beyond nuclear densities the medium is expected to undergo a phase transition to a state of color superconducting, in which quarks become correlated in Cooper-pairs analog to metal superconductors [18].

The QGP phase diagram is experimentally only accessible via heavy-ion collision experiments, where sufficiently high temperatures and densities can be reached to produce the QGP. Depending on the collision energy in these experiments, different regions and trajectories in the QGP phase diagram can be explored. The collision energy is usually expressed using the center of mass energy per nucleon pair $\sqrt{s_{NN}}$ of the collision. Higher collision energies are connected to a higher initial temperature of the medium and a smaller baryon chemical potential, whereas smaller energies are suited to study the phase diagram at a larger baryon chemical potential. Many heavy-ion collision experiments have been or are performed to study the QGP phase. Examples of the low-energy collision experiments are, among many others, FOPI or HADES at GSI, whereas experiments for example at RHIC or the LHC mostly probe the QGP at large collision energies. Usually, in these experiments, the beam energy is varied to search

for the critical point. The exploration of the QCD phase diagram is still an active field of research, new regions of it will be also investigated in upcoming heavy-ion facilities like NICA and FAIR at large baryon chemical potential [15].

2.3 Heavy-Ion Collisions

Ultra-relativistic heavy-ion collisions can be used to study the state of the QGP because they provide large enough temperatures and energy densities to reach this phase. For heavy-ion collisions at the LHC, which will be studied here, the initial temperature of the medium exceeds the critical temperature T_c, while the baryon chemical potential is approximately zero, conditions that are similar to the ones in the early universe. However, since the state of the QGP in heavy-ion collision experiments exists only for a very short time of about 10^{-23} s, it is impossible to perform any direct measurements in this phase. Only later stages of the evolution of the collision system can be observed in the detectors. Because of this, an understanding of the full evolution of a heavy-ion collision is necessary to infer information about the QGP. The space-time evolution of a collision can be divided into several stages, which are depicted in fig. 2.4. For the evolution it is convenient to introduce a proper time scale $\tau = \sqrt{t^2 - z^2}$, accounting for relativistic effects such that it is invariant under Lorentz boosts and the space-time rapidity $\eta_s = \frac{1}{2} \ln \left(\frac{t+z}{t-z} \right)$ [19].

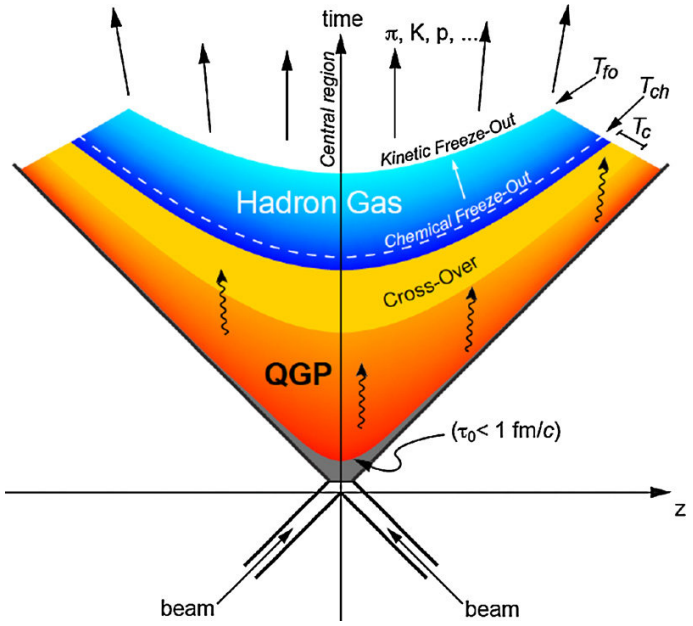


FIGURE 2.4: The space-time diagram for the evolution of a heavy-ion collision. Taken from [20].

Using this definition, the following stages are defined, mainly adapted from [20]:

- **Initial state** At time $t = \tau = 0$, the two Lorentz contracted nuclei collide with ultra-relativistic velocity. The initial state is defined by the positions and interactions of all the constituent nucleons of the nuclei.
- **Pre-equilibrium** In the pre-equilibrium phase hard scattering processes with large transferred momenta between the interacting partons may occur, which lead to the production of a few highly energetic and massive particles, also called hard probes. However, this stage is mostly dominated by strong interaction dynamics

between the constituent particles that drive the system to approximate thermalization. The exact dynamic in this stage is not well understood yet.

- **Thermal equilibrium and QGP phase** After a proper time of $\tau_0 \lesssim 1\,\mathrm{fm/c}$, the medium approaches the state of a local thermal equilibrium (the QGP), which can be described well within the theory of relativistic hydrodynamics [21]. The strong microscopic dynamics lead to rather small dissipative transport coefficients, such that the QGP is an almost perfect fluid. Because of the large temperature and density, the QGP is expanding rapidly, while cooling down.

- **Hadronization** When the system passes the critical temperature T_c, the QGP starts to hadronize, and quarks and gluons are confined into hadrons. If there is still a large enough density, there are many inelastic scattering events, such that the chemical and kinetic equilibrium is maintained.

- **Chemical freeze-out** At some point, the temperature of the medium becomes so low, that inelastic scattering events cease and the chemical equilibrium is no longer maintained. This process is called chemical freeze-out and the particle yields are fixed at this point up to decays of particles with very short lifetimes. The temperature defining the freeze-out is the chemical freeze-out temperature T_{ch} and it has been argued that it is near the critical temperature T_{c} [22].

- **Kinetic freeze-out** If the temperature drops even further, also elastic scatterings cease and particle spectra and correlations are fixed. This is the kinetic freeze-out, which occurs at temperatures of $T_{\mathrm{kin}} \sim 100\,\mathrm{MeV}$.

Ultimately, what can be observed in the detectors are the particle spectra and correlations that arise from the evolution of the system. The particles can originate from different stages of the evolution. While high energetic particles and heavy quarks come mainly from the pre-equilibrium phase, light hadrons originate from the hadronization process.

Chapter 3

Modeling of heavy-ion collisions

From the whole evolution of a heavy-ion collision, only the free-streaming particles from the last stage can be measured in the detectors. All the other stages of the evolution are not directly accessible by the experiments, such that their physical properties cannot be measured. However, they still have a large impact on the resulting particle spectra and correlations, such that it can be attempted to infer them indirectly. This can be achieved by modeling the whole evolution process of a heavy-ion collision and comparing the results of this modeling to experimental data. The established approach in the field of heavy-ion collisions is to model every stage of the evolution (as given in sec. 2.3) individually. The description of each stage is then matched to the others to obtain one complete model for the full evolution. The physical properties of the stages can then be inferred by optimizing the model parameters such that the model output matches the experimental data. This approach has already been used successfully multiple times [23], [24], however, quantitative estimates for important properties of the QGP are still not very precise. One reason for this is that the physics of some stages of the evolution is not fully understood yet and therefore different physical assumptions can be considered. The goal of this thesis is to contribute to a better understanding of the properties of the QGP and to provide quantitative estimates for these properties by applying such modeling. The difference with respect to other analyses is that a newly developed hydrodynamic software package called Fluidum [7] will be used to model the stage of the expansion of the QGP. Usually, modeling is mostly based on event-by-event hydrodynamics, where individual heavy-ion collisions are simulated and the QGP properties are inferred by averaging over thousands of events. This approach is computationally expensive but offers the opportunity to also study high-order observables like flow coefficients and multi-particle correlations. In contrast, Fluidum is based on averaging the initial event characterization and evolving only this average hydrodynamically, therefore it is computationally much cheaper which is advantageous for a quantitative analysis. In the following, the modeling of all stages of the heavy-ion collision that will be used in this thesis will be laid out.

3.1 Initial conditions

Initial condition models provide a description for the entropy or energy density at thermalization time τ_0, the time at which the QGP is established and the fluid-dynamic description becomes valid. There are two conceptually different approaches to modeling the initial state. In the static approach, it is assumed that the energy or entropy after the collision remains unchanged in space-time until the time τ_0, such that modeling its density directly after the collision provides also the initial conditions for the hydrodynamical evolution at time τ_0. In contrast to this, in the dynamical approach, the dynamics of the constituent particles in the pre-equilibrium stage are explicitly modeled in addition, to describe the situation more realistically. Both approaches have been used successfully in the past and many different models for characterizing the initial state exist [25]. In this thesis, the non-dynamical Trento model [8], which is connected to the more general Glauber model, will be used to produce the initial conditions for the

hydrodynamical evolution, thus omitting a specific description of the pre-equilibrium stage.

3.1.1 Event characterization

Before the initial condition model is introduced, it is convenient to characterize the geometry of a collision event. Individual heavy-ion collision events can differ significantly in entropy production and in the number of produced particles (the so-called multiplicity) depending on the configuration of the individual collision. Consider the collision of two nuclei A and B, as it is depicted in fig. 3.2.

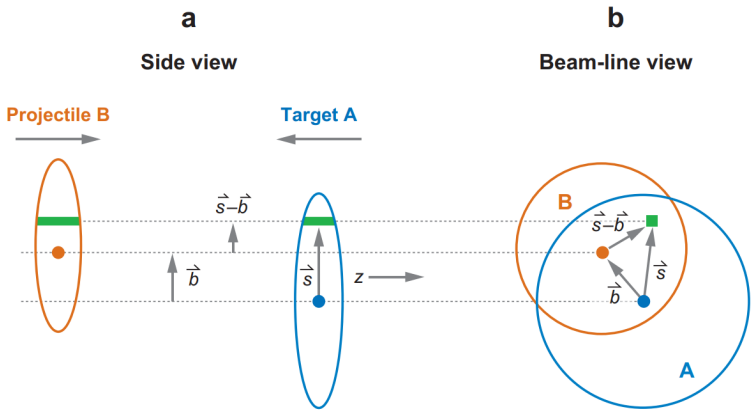


FIGURE 3.1: The transverse and longitudinal geometry of a collision. Taken from [26].

The impact parameter $\vec{b}$ is defined as the vector, that connects the two centers of the colliding nuclei in the plane transverse to the beam. The events can be can then be classified into *centrality classes*. This can be done by defining percentiles for the hadronic cross-section σ. This means, that the five percent of collisions with the largest cross-section correspond to the 5% centrality class, the next 5% to the 5-10% centrality class, and so on. However, the cross-section cannot be measured in the experiment directly, therefore the centrality classes are usually defined using observables that are highly correlated with it. In the ALICE experiment centrality is for example defined by the energy deposited in the V0 detectors [1]. The quantity that will be used in this thesis to define the centrality is the charged particle multiplicity, which is negatively correlated with the impact parameter $\vec{b}$. Therefore the five percent of collisions with the largest multiplicity correspond to the 5% centrality class and so on. Mathematically, this can be expressed by [26]

$$\frac{\int_{\infty}^{n_i} (dN_{evt}/dN_{ch})\, dN_{ch}}{\int_{\infty}^{0} (dN_{evt}/dN_{ch})\, dN_{ch}} = i, \tag{3.1}$$

where n_i is the boundary for percentile i and dN_{evt}/dN_{ch} is the charged-particle multiplicity. This definition can also be used to classify events of initial condition models as it will be done in the following.

3.1.2 Trento

Trento [8] is an effective initial conditions model to generate realistic Monte Carlo entropy profiles without assuming specific physical mechanisms for entropy production, pre-equilibrium dynamics, or thermalization [8]. The model arises from simple geometric and statistical considerations. Its main idea is to describe the collision of two nuclei

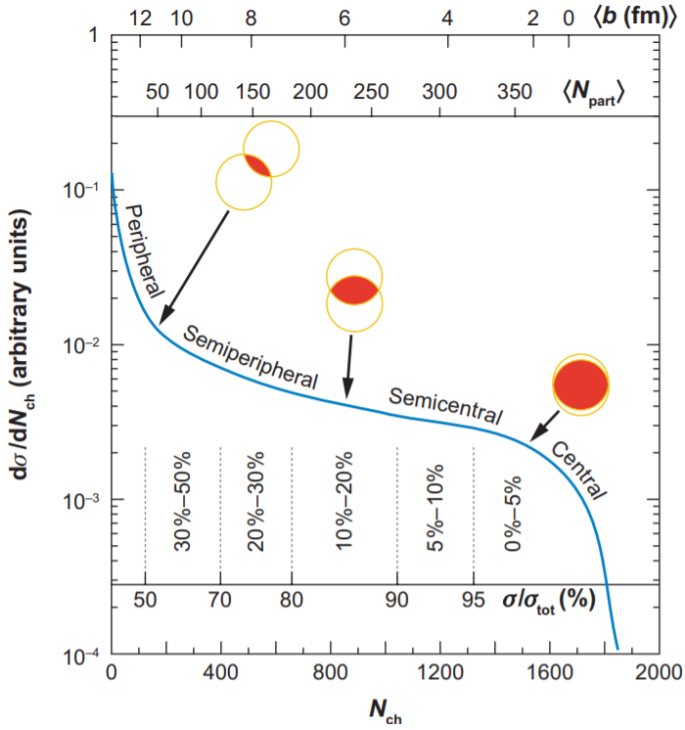


FIGURE 3.2: Illustration of the centrality classes related to different observables. Taken from [26].

by the individual collisions of their constituent nucleons and then relate the arising distribution to the production of entropy. To describe the collision in terms of nucleon interactions, at first, the positions of the nucleons inside the nuclei have to be specified. This can be done by regarding the nuclear density within a nucleus, which is often characterized by a Woods-Saxon distribution in terms of the nuclear density at the core of the nucleus ρ_0, the "nuclear" radius R, which defines the radius where the density is halved ($\rho(R) = \rho_0/2$), and the surface thickness a of the nucleus:

$$\rho(r) = \frac{\rho_0}{1 + \exp[(r - R)/a]} \tag{3.2}$$

For individual Trento events, the position of the nucleons inside a nucleus is sampled from this distribution. The density of each nucleon can be described by a Gaussian distribution with some width w

$$\rho_{\text{nucleon}}(x, y, z) = \frac{1}{2\pi w^2} \exp\left(-\frac{x^2 + y^2 + z^2}{2w^2}\right). \tag{3.3}$$

Using the positions of the nucleons within the nuclei and the density distribution for the nucleons, the collision probability for the collision of two individual nucleons A and B can be calculated by

$$P_{\text{coll}} = 1 - \exp\left[-\sigma_{gg} \int dxdy \int dz\rho_A \int dz\rho_B\right], \tag{3.4}$$

where σ_{gg} is an effective parton-parton cross-section adjusted such that the total cross-section is equal to the experimental inelastic nucleon-nucleon cross-section. When sampling from this distribution, the number of participants in the collision can be determined. To describe the deposition of entropy, the so-called *thickness* function can be

introduced, which is the density of the nucleons projected to the transverse plane orthogonal to the beam direction. Mathematically this manifests in integrating out the z coordinate from the density:

$$T_{\mathrm{A,B}}(x,y) = w_{\mathrm{A,B}} \int dz \rho_{A,B}(x,y,z). \tag{3.5}$$

The additional weights $w_{\mathrm{A,B}}$ are introduced to account for multiplicity fluctuations observed in the experiment, therefore eq. 3.5 gives rise to the *fluctuated thickness* function. The weights are sampled in Trento from a gamma distribution.

$$P_k(w) = \frac{k^k}{\Gamma(k)} w^{k-1} e^{-kw} \tag{3.6}$$

The shape parameter k adjusts the magnitude of the fluctuations: values between zero and one lead to large multiplicity fluctuations, whereas values of $k >> 1$ suppress them. The thickness functions of the two colliding nuclei can be constructed by adding up the contributions from all individual nucleons

$$T_{\mathrm{A,B}} = \sum_{i=1}^{N_{\mathrm{part}}} w_i \int dz \rho_{\mathrm{nucleon}} \left(x - x_i, y - y_i, z - z_i\right). \tag{3.7}$$

The core assumption of the Trento model is, that the overlap of the thickness functions of the two nuclei T_{A} and T_{B} is related to the production of entropy. This relation is assumed to be described by a scalar field $f\left(T_A, T_B\right)$, such that f is proportional to the entropy created at midrapidity and at the thermalization time τ_0:

$$f \propto dS/\ dy|_{\tau=\tau_0} \tag{3.8}$$

For the function f there are several reasonable choices with different physical interpretations. In Trento the *reduced thickness* function is given by

$$f = T_R\left(p; T_{\mathrm{A}}, T_{\mathrm{B}}\right) \equiv \left(\frac{T_A^p + T_B^p}{2}\right)^{1/p}, \tag{3.9}$$

where the dimensionless continuous parameter p is used to interpolate between different physical mechanisms for entropy production. By adjusting the p parameter, the function interpolates between the minimum and maximum value of T_{A} and T_{B}, passing arithmetic, geometric and harmonic means for certain values

$$T_R = \begin{cases} \max\left(T_A, T_B\right) & p \to +\infty, \\ \left(T_A + T_B\right)/2 & p = +1, \quad \text{(arithmetic)} \\ \sqrt{T_A T_B} & p = 0, \quad \text{(geometric)} \\ 2T_A T_B/\left(T_A + T_B\right) & p = -1, \quad \text{(harmonic)} \\ \min\left(T_A, T_B\right) & p \to -\infty. \end{cases} \tag{3.10}$$

Using the procedure above, entropy profiles at thermalization time can be produced for heavy-ion collisions. Up to a normalization factor, these profiles can be taken as initial conditions for the further evolution of the system. This evolution, which is done using a hydrodynamic approach will be laid out in the next section.

3.2 Fluid dynamic evolution of the QGP

After the initial collision and the subsequent pre-equilibrium phase, the heavy-ion collision system thermalizes and the QGP forms. As mentioned before, the evolution of

the QGP can be modeled by relativistic fluid dynamics, also loosely referred to as hydrodynamics. Within this approach, the complex microscopic dynamics of the particles in the system are summarized in a few macroscopic, thermodynamic and transport properties. However, the validity of this approach relies on the assumption that local thermal equilibrium is established and maintained during the time of the evolution of the QGP until the freeze-out phase. This assumption is quite strong, but experimental results and the success of fluid dynamic modeling suggest that it holds [27]. In the past, QCD dynamics have been studied already extensively in the context of fluid dynamics, nevertheless, its thermodynamic and transport properties are not fully understood and quantified yet. An overview of the developments in this field can be found in [25], [27], [28]. In the following, the basic aspects of relativistic fluid dynamics will be introduced as well as the software package Fluidum [7], which will be used for hydrodynamic modeling of the quark-gluon plasma in this thesis. This section mainly follows [27] and [7]. All calculations will be expressed in natural units ($c = \hbar = k_B = 1$).

3.2.1 Ideal fluid dynamics

Ideal fluid dynamics is a simplified description of real fluids where the fluid is assumed to be incompressible and non-viscous, which means that it has no internal resistance to the fluid flow (viscosity). An important assumption of ideal fluid dynamics is that the fluid is in local thermodynamic equilibrium, meaning that each infinitesimal fluid element is in thermodynamic equilibrium with its neighboring elements. The state of the fluid may then be defined by certain densities and currents associated with conserved quantities which are mathematically expressed in terms of continuous scalar or vector fields. For fluid dynamics, the most important conserved quantities are energy and momentum. There exist other conserved charges e.g. baryon number, strangeness, and electric charge, however, these will not be considered here. For relativistic fluids, the densities and currents are encoded in the energy-momentum tensor $T^{\mu\nu}$. In the local rest frame of the fluid, where the fluid velocity $\vec{v}$ is zero everywhere, it can be expressed for the ideal fluid in of the energy density $\epsilon(x)$ and pressure $p(x)$ by

$$T^{\mu\nu} = \begin{pmatrix} \epsilon(x) & 0 & 0 & 0 \\ 0 & p(x) & 0 & 0 \\ 0 & 0 & p(x) & 0 \\ 0 & 0 & 0 & p(x) \end{pmatrix}. \tag{3.11}$$

To describe the fluid in the global rest frame, the four-velocity field $u^\mu(x)$ has to be taken into account, which defines the velocity at each space-time point and is given by

$$u^\mu(x) \equiv \gamma \begin{pmatrix} 1 \\ \vec{v} \end{pmatrix}, \tag{3.12}$$

where $\gamma = 1/\sqrt{1-\vec{v}^2}$ and $u^\mu(x)$ is normalized such that

$$u^2 \equiv g_{\mu\nu} u^\mu(x) u^\nu(x) = -1, \tag{3.13}$$

using the metric $g_{\mu\nu} = \mathrm{diag}(-1, +1, +1, +1)$ in Minkowski space with Euclidean coordinates. The energy-momentum tensor in the global rest frame can now be obtained by Lorentz boosting the tensor of the local rest frame according to the four-velocity field $u^\mu(x)$. This procedure leads to an energy-momentum tensor of the following form

$$T^{\mu\nu} = \varepsilon u^\mu u^\nu + p\left(g^{\mu\nu} + u^\mu u^\nu\right). \tag{3.14}$$

The dynamics of a relativistic fluid can be inferred by considering the local conservation of energy and momentum, which can be expressed mathematically in terms of the derivative of the energy-momentum tensor by

$$\nabla_\mu T^{\mu\nu} = 0, \tag{3.15}$$

where $\nabla_\mu \equiv \partial/\partial x^\mu$ transforms as a covariant derivative under Lorentz transformations. Utilizing these conservation equations, the equations of motion of ideal fluid dynamics can be constructed. They describe the full dynamic of the ideal relativistic fluid and are given by the partial differential equations

$$u^\mu \partial_\mu \epsilon + (\epsilon + p)\nabla_\mu u^\mu = 0, \tag{3.16}$$
$$(\epsilon + p)u^\mu \nabla_\mu u^\nu + \left(g^{\nu\mu} + u^\nu u^\mu\right)\partial_\mu p = 0. \tag{3.17}$$

As it is apparent in eqs. 3.16 and 3.17, three fields, namely ϵ, p and u^μ, are required to describe the perfect fluid. They correspond to five degrees of freedom. Since the conservation laws only provide four equations, additionally the Equation of State, $p = p(\epsilon)$, has to be taken into account to form a closed system of equations, that can then be solved to obtain the dynamics if initial conditions for ϵ, p and u^μ are provided.

3.2.2 Dissipative fluid dynamics

To describe a fluid more realistically, dissipative corrections to the stress-energy tensor must be taken into account. In this way, heat and energy may dissipate by friction between the fluid elements in the fluid. To account for the dissipation effects, the energy-momentum tensor of eq. 3.14 has to be modified by the term $\Pi^{\mu\nu}$, containing gradients of thermodynamic quantities e.g. the fluid velocity [29]

$$T^{\mu\nu} = T^{\mu\nu}_{\text{ideal}} + \Pi^{\mu\nu}. \tag{3.18}$$

The energy-momentum tensor may then be decomposed, such that

$$T^{\mu\nu} = \epsilon u^\mu u^\nu + (p + \pi_{\text{bulk}})\,\Delta^{\mu\nu} + \pi^{\mu\nu}, \tag{3.19}$$

where $\Delta^{\mu\nu} = g^{\mu\nu} + u^\mu u^\nu$ is the projector into the space components of the fluid rest frame, $\pi^{\mu\nu}$ is the symmetric shear stress tensor, which is traceless ($\pi^\mu{}_\mu = 0$) and orthogonal to the fluid velocity ($u_\mu \pi^{\mu\nu} = 0$), and π_{bulk} is the bulk viscous pressure. Analogously to the ideal case, the equations of motion can be derived from the conservation laws eq. 3.15, which yields

$$u^\mu \partial_\mu \epsilon + (\epsilon + p + \pi_{\text{bulk}})\,\nabla_\mu u^\mu + \pi^{\mu\nu}\nabla_\mu u_\nu = 0, \tag{3.20}$$
$$(\epsilon + p + \pi_{\text{bulk}})\,u^\nu \nabla_\nu u^\mu + \Delta^{\mu\nu}\partial_\nu (p + \pi_{\text{bulk}}) + \Delta^\mu{}_\nu \nabla_\rho \pi^{\rho\nu} = 0. \tag{3.21}$$

In the first order in derivatives with respect to the fluid velocity, the shear stress tensor $\pi^{\mu\nu}$ and the bulk viscous pressure π_{bulk} can be approximated by

$$\pi_{\text{bulk}} = -\zeta \nabla_\mu u^\mu \tag{3.22}$$
$$\pi^{\mu\nu} = -2\eta \left(\frac{1}{2}\Delta^{\mu\alpha}\Delta^{\nu\beta} + \frac{1}{2}\Delta^{\mu\beta}\Delta^{\nu\alpha} - \frac{1}{3}\Delta^{\mu\nu}\Delta^{\alpha\beta}\right)\nabla_\alpha u_\beta, \tag{3.23}$$

where the bulk viscosity $\zeta(\epsilon)$ and the shear viscosity $\eta(\epsilon)$ are introduced, which are so-called transport coefficients as they describe the microscopic momentum exchange and therefore the dissipation of energy. This procedure follows the relativistic generalization of the Navier-Stokes theory. The non-relativistic Navier-Stokes equation can be recovered by regarding eq. 3.21 in the non-relativistic limit. Although this procedure is the straightforward generalization of the non-relativistic case, the relativistic formulation of Navier-Stokes theory has been shown to violate the relativistic causality principle and to be linearly unstable [30]. Therefore, second-order theories have been introduced to overcome these problems. The idea of these theories is to provide dynamical equations for the shear stress tensor $\pi^{\mu\nu}$ and the bulk viscous pressure π_{bulk}. For Fluidum, the evolution equations for the shear stress tensor $\pi^{\mu\nu}$ and the bulk viscous pressure π_{bulk} are given by [7]

$$\begin{aligned} P^{\mu\rho}_{\nu\sigma}\Big[\tau_{\text{shear}}\left(u^\lambda\nabla_\lambda\pi^\sigma_\rho - 2\pi^{\sigma\lambda}\omega_{\rho\lambda}\right) + 2\eta\nabla_\rho u^\sigma - \varphi_7\pi^\lambda_\rho\pi^\sigma_\lambda + \tau_{\pi\pi}\pi^\sigma_\lambda\sigma^\lambda_\rho \\ -\lambda_{\pi\Pi}\pi_{\text{bulk}}\nabla_\rho u^\sigma\Big] + \pi^\mu_\nu\left[1 + \delta_{\pi\pi}\nabla_\rho u^\rho - \varphi_6\pi_{\text{bulk}}\right] = 0 \end{aligned} \tag{3.24}$$

$$\begin{aligned} \tau_{\text{bulk}}\,u^\mu\partial_\mu\pi_{\text{bulk}} + \pi_{\text{bulk}} + \zeta\nabla_\mu u^\mu + \delta_{\Pi\Pi}\pi_{\text{bulk}}\nabla_\mu u^\mu - \varphi_1\pi^2_{\text{bulk}} \\ - \lambda_{\Pi\pi}\pi^{\mu\nu}\nabla_\mu u_\nu - \varphi_3\pi^\mu_\nu\pi^\nu_\mu = 0, \end{aligned} \tag{3.25}$$

where the projector $P^{\mu\rho}_{\nu\sigma}$ to the symmetric, transverse, and traceless part of a tensor is defined as

$$P^{\mu\nu}_{\rho\sigma} = \frac{1}{2}\Delta^\mu_\rho\Delta^\nu_\sigma + \frac{1}{2}\Delta^\mu_\sigma\Delta^\nu_\rho - \frac{1}{3}\Delta^{\mu\nu}\Delta_{\rho\sigma} \tag{3.26}$$

and the abbreviations

$$\sigma_{\mu\nu} = P^{\rho\sigma}_{\mu\nu}\nabla_\rho u_\sigma, \quad \omega_{\mu\nu} = \frac{1}{2}\left(\nabla_\mu u_\nu - \nabla_\nu u_\mu\right) = \frac{1}{2}\left(\partial_\mu u_\nu - \partial_\nu u_\mu\right) \tag{3.27}$$

were used. Within eqs. 3.24 and 3.25, various transport coefficients are introduced. The first order coefficients η and ζ for the shear and bulk viscosities were already introduced and have the largest impact on the dynamics. From the second order transport coefficients, the relaxation times τ_{shear} and τ_{bulk} also have a considerable impact on the dynamics. They quantify how fast the shear stress tensor and the bulk viscous pressure return to their asymptotic values $\pi^{\mu\nu} = -2\eta\sigma^{\mu\nu}$ and $\pi_{\text{bulk}} = -\zeta\nabla_\rho u^\rho$ [7]. Other second-order transport coefficients like $\tau_{\pi\pi}$, $\delta_{\pi\pi}$, $\lambda_{\pi\Pi}$, $\delta_{\Pi\Pi}$, $\lambda_{\Pi\pi}$, φ_7, φ_6, φ_1 and φ_3 have only a minor impact. Equations 3.20, 3.21, 3.24 together with 3.25 form a closed system of partial differential equations, which can be solved numerically to obtain the dynamics of the system.

3.2.3 Transport coefficients

In the context of dissipative relativistic fluid dynamics, transport coefficients appear. Since these are important properties of the fluid, they will be laid out here in more detail. In general in physics, transport coefficients measure how rapidly a system returns to equilibrium after it has been perturbed. For the perturbed system, gradients of physical properties arise, which lead to a flux bringing the system back to equilibrium by transporting the physical quantity in an irreversible process. The transport coefficients γ_k of physical quantity k occur as the connection between the flux J_k and the gradient force X_k in transport laws

$$J_k = \gamma_k X_k. \tag{3.28}$$

Such transport coefficients are for example the thermal conductivity, which describes the conduction of heat and therefore the transport of energy, or the viscosity, which is an example of microscopic transport of momentum by the irreversible process of friction. As elaborated in the last section, the most important transport coefficients in dissipative relativistic fluids like the QGP are the shear and bulk viscosity as well as the relaxation times τ_{shear} and τ_{bulk}. The shear viscosity η is the diffusion coefficient of momentum transfer perpendicular to the local velocity of the fluid. It acts against the buildup of flow anisotropies between fluid layers. On the other hand the bulk viscosity ζ acts against the buildup of radial flow, therefore damping the expansion of the fluid. In general, the determination of these transport properties is difficult in QCD. They can be obtained in very weakly interacting theories from perturbation theory and in very strongly interacting theories via the AdS/CFT correspondence [31]. For theories that are located in between these extremes, the determination is still an open problem. For strongly interacting systems a lower bound was postulated for the ratio of the shear viscosity to entropy density η/s [32], which is given by

$$\frac{\eta}{s} \geq \frac{1}{4\pi}. \tag{3.29}$$

It can be shown that the shear and bulk viscosities always appear in the hydrodynamic equations of motion in the dimensionless combinations η/s and ζ/s with the entropy density s [29]. Therefore, the strength of the shear and bulk viscosities is often expressed in terms of these dimensionless quantities and so it will be in this thesis. Following the second law of thermodynamics, it can be also derived, that the inequalities $\eta/s \geq 0$ and $\zeta/s \geq 0$ hold [29].

3.2.4 Fluidum

To model the evolution of the QGP, the equations of motion that arise from the treatment of the dissipative relativistic fluid have to be solved. Since these partial differential equations are not solvable analytically, numerical methods have to be applied. One of these numerical methods is the recently developed software package Fluid dynamics of heavy-ion collisions with Mode expansion (Fluidum [7]), which evolves the fluid fields numerically according to the equations of motions, starting from provided initial conditions. In the following, the reasoning behind Fluidum will be laid out on the basis of [7].

To solve the equations of motions numerically, a coordinate system for the collision system has to be chosen. In Fluidum, its origin is set at the collision point in the center of the expanding fireball. The z-axis and the laboratory time t are expressed using the proper time $\tau = \sqrt{t^2 - z^2}$ and the rapidity $\eta = \operatorname{arctanh}(z/t)$, such that $t = \tau \cosh(\eta)$ and $z = \tau \sinh(\eta)$. In the transverse plane, the coordinates x and y are conveniently expressed in cylindrical coordinates, such that $r = \sqrt{x^2 + y^2}$ and $\phi = \arctan(y/x)$. The reason for the use of this coordinate system is that it is particularly well suited to describe the approximate azimuthal rotation symmetry and the approximate longitudinal rapidity boost symmetry observed in real heavy-ion collisions. Using this coordinate system, every space-time point can be described by τ, r, ϕ, and η.

After introducing the coordinate system, the scheme for solving the equations of motion can be described. For this, the general case of a set of hyperbolic, quasi-linear

partial differential equations is considered. This set of hyperbolic equations can be expressed symbolically by

$$\begin{aligned}\mathbf{A}(\mathbf{\Phi},\tau,r)\cdot\partial_\tau\mathbf{\Phi}+\mathbf{B}(\mathbf{\Phi},\tau,r)\cdot\partial_r\mathbf{\Phi}+\mathbf{C}(\mathbf{\Phi},\tau,r)\cdot\partial_\phi\mathbf{\Phi}\\+\mathbf{D}(\mathbf{\Phi},\tau,r)\cdot\partial_\eta\mathbf{\Phi}-\mathbf{S}(\mathbf{\Phi},\tau,r)=0,\end{aligned}\tag{3.30}$$

where $\mathbf{\Phi}$ is the "Nambu spinor" with N components which contains all fluid fields needed to describe the state of the fluid, $\mathbf{A},\mathbf{B},\mathbf{C}$ and $\mathbf{D}$ are $N\times N$ coefficient matrices and the source term $\mathbf{S}$ is a N-component vector. In the case of a relativistic dissipative fluid, these may be for example the temperature, the fluid velocity, the independent components of the shear stress, and the bulk viscous pressure. The coefficients in eq. 3.30 can be identified by comparing to the equations of motions, thus relating the specific problem to the general case. The main idea behind Fluidum is now that $\mathbf{\Phi}$ can be split into a background part that is symmetric under rotations and Lorenz-boosts in the z direction and a symmetry breaking part accounting for deviations with respect to the symmetric part

$$\mathbf{\Phi}(\tau,r,\phi,\eta)=\mathbf{\Phi}_0(\tau,r)+\epsilon\mathbf{\Phi}_1(\tau,r,\phi,\eta).\tag{3.31}$$

The background field $\mathbf{\Phi}_0(\tau,r)$ can be seen as a statistical expectation value when averaging over a large number of collision events, while $\mathbf{\Phi}_1(\tau,r,\phi,\eta)$ can be seen as the fluctuations of each individual event. The advantage of this approach becomes clear when the standard way to solve the evolution of the QGP system is considered. Usually, the collisions are simulated event-by-event, which means that for each collision an initial entropy density is generated and then evolved hydrodynamically by solving the equations of motions. The mean behavior of the system is then extracted by averaging over all simulated events. This approach is computationally expensive because of the individual simulation of thousands of events and therefore limited in its ability to obtain precise quantitative results. In contrast to this, the averaging in Fluidum is done before solving the equations of motions with the initial conditions, thus the evolution has to be performed only once to infer the mean evolution behavior. It is therefore computationally a lot cheaper. To solve the partial differential equations within this approach, eq. 3.31 can be inserted into eq. 3.30, such that

$$\begin{aligned}&\mathbf{A}\left(\mathbf{\Phi}_0+\epsilon\mathbf{\Phi}_1,\tau,r\right)\cdot\partial_\tau\left(\mathbf{\Phi}_0+\epsilon\mathbf{\Phi}_1\right)+\mathbf{B}\left(\mathbf{\Phi}_0+\epsilon\mathbf{\Phi}_1,\tau,r\right)\cdot\partial_r\left(\mathbf{\Phi}_0+\epsilon\mathbf{\Phi}_1\right)\\&+\mathbf{C}\left(\mathbf{\Phi}_0+\epsilon\mathbf{\Phi}_1,\tau,r\right)\cdot\partial_\phi\left(\mathbf{\Phi}_0+\epsilon\mathbf{\Phi}_1\right)+\mathbf{D}\left(\mathbf{\Phi}_0+\epsilon\mathbf{\Phi}_1,\tau,r\right)\cdot\partial_\eta\left(\mathbf{\Phi}_0+\epsilon\mathbf{\Phi}_1\right)\\&-\mathbf{S}\left(\mathbf{\Phi}_0+\epsilon\mathbf{\Phi}_1,\tau,r\right)=0.\end{aligned}\tag{3.32}$$

If the fluctuation fields $\mathbf{\Phi}_1(\tau,r,\phi,\eta)$ are small with respect to the background field, this equation can be expanded in terms of a small expansion parameter ϵ. The equations of motion for the background field can then be obtained by considering the zeroth order in epsilon, which is given by

$$\mathbf{A}_0\left(\mathbf{\Phi}_0,\tau,r\right)\cdot\partial_\tau\mathbf{\Phi}_0(\tau,r)+\mathbf{B}_0\left(\mathbf{\Phi}_0,\tau,r\right)\cdot\partial_r\mathbf{\Phi}_0(\tau,r)-\mathbf{S}_0\left(\mathbf{\Phi}_0,\tau,r\right)=0,\tag{3.33}$$

where only the coordinates τ and r are relevant. $\mathbf{\Phi}_0$ is introduced since fewer fields are needed to characterize the fluid because of the symmetry constraints for the background field, therefore $\mathbf{\Phi}_0$ contains fewer independent components than $\mathbf{\Phi}$. $\mathbf{A}_0$ and $\mathbf{B}_0$ are similarly reduced matrices, which can be seen as projections of $\mathbf{A}$ and $\mathbf{B}$ to the reduced space of independent components. Carrying out the same procedure for the perturbation field, it is also possible to infer the equations of motions for the first or

higher orders of the perturbation field. This will not be discussed here, detailed information about the treatment of higher orders is given in [7].
By assigning the equations of motion of the relativistic dissipative fluid to the symbolic equations above, sets of hyperbolic partial differential equations for the evolution of the background and perturbation field arise. To solve these numerically, the radial coordinate r is discretized and the resulting differential equation is solved within Fluidum via a pseudo-spectral method, where the solution is approximated as a linear superposition of certain basis functions. The exact numerical scheme will not be laid out here, it can be found together with its validation in [7].

3.3 Hadronization and resonance decays

At some point in the evolution of the QGP, the system passes the cross-over temperature T_c, and the quarks get confined in hadrons. This process is known as *Hadronization*. To model this phase, the continuous fluid fields of the hydrodynamic evolution of the QGP have to be related to the production of discrete particles such that momenta and energy are conserved. This is usually done based on the Cooper-Frye procedure [33]. Within this approach, the transition to particles is described by a freeze-out temperature T_{freeze} which defines a freeze-out hyper-surface σ in the space-time evolution. On the freeze-out surface, the fluid fields are converted to particles using the Cooper-Frye formula, which performs an integration over it to infer particle momentum spectra. It is given by [9]

$$E_{\mathbf{p}} \frac{dN_a}{d^3\mathbf{p}} = \frac{\nu_a}{(2\pi)^3} \int_\sigma f_a \left(-u^\nu p_\nu, T, \mu\right) p^\mu d\sigma_\mu, \tag{3.34}$$

where ν_a accounts for the spin degeneracy of the particle and f_a is the particle distribution function of particle species a, which is dependent on the fluid temperature $T(x)$, the fluid velocity $u^\mu(x)$ and the chemical potential $\mu(x)$ for the ideal case. $\frac{dN_a}{d^3\mathbf{p}}$ is the resulting momentum spectrum of species a and $E_{\mathbf{p}} = \sqrt{m^2 + \mathbf{p}^2}$ its energy. The particle distribution function f_a is given for an ideal fluid as the equilibrium Bose-Einstein or Fermi-Dirac distribution f_{eq} depending on the species. However, for the viscous case, the distributions have to be corrected due to bulk and shear viscous dissipation, such that the particle distribution function also depends on the viscous shear-stress tensor $\pi^{\mu\nu}(x)$ and the bulk viscous pressure $\pi_{\text{bulk}}(x)$. The functional dependence of the distribution function on viscous corrections is an unresolved problem, even at linear order in dissipative terms [9]. Here, the following parametrization of the corrections in linear order will be used for the particalization:

$$f = f_{\text{eq}} + \delta f^{\text{bulk}} + \delta f^{\text{shear}}, \tag{3.35}$$

with the corrections [24]

$$\begin{aligned} \delta f^{\text{bulk}} &= f_{\text{eq}} \left(1 \pm f_{\text{eq}}\right) \left[\frac{\bar{E}_p}{T}\left(\frac{1}{3} - c_s^2\right) - \frac{m^2}{3TE_p}\right] \frac{\pi_{\text{bulk}}}{\zeta/\tau_{\text{bulk}}}, \\ \delta f^{\text{shear}} &= f_{\text{eq}} \left(1 \pm f_{\text{eq}}\right) \frac{\pi_{\rho\nu} p^\rho p^\nu}{2(\epsilon + p)T^2}, \end{aligned} \tag{3.36}$$

where $c_s(T)$ is the speed of sound of the medium at T_{freeze}, m is the mass of the primary resonance, and τ_{bulk}/ζ is the ratio of the bulk relaxation time and the bulk viscosity. With this procedure, the fluid fields are converted to particles with certain momenta.
What has not been considered yet are decays of short-lived particles (so-called resonances), which contribute significantly to the particle spectra of particles that reach the

detectors. To take into account resonance decays, all sufficiently unstable particles that are produced by the Cooper-Frye procedure have to be decayed according to their decay channels to lighter stable particles. Often the decay processes are simulated by Monte-Carlo generators [34] or by semi-analytic treatments of the decay integrals [35]. This can be computationally expensive since large cascades of decays may have to be considered. Therefore, in this thesis, a program called FastReso [9] will be used to treat resonance decays. It is computationally more efficient since the particle decays are not computed event-by-event but decay maps are pre-computed. It will be explained in more detail in the next section, but before a second freeze-out temperature will be introduced into the Cooper-Frye procedure.
Until now, only one freeze-out temperature T_{freeze} has been considered. However, in heavy-ion collisions two freeze-outs, the chemical and kinetic freeze-out, occur. To account for this, the aforementioned procedure can be modified by introducing a phase of partial chemical equilibrium. The temperature T_{freeze} defining the freeze-out surface of the fluid can be identified with T_{chem} since at this point the abundances of quasi-particles are fixed. The inelastic scattering events cease, such that the chemical equilibrium cannot be maintained, but elastic scattering still occurs and impacts the momentum distributions of the particles. To describe the evolution in this phase, it is convenient to define conservation laws. Excluding resonance decays, the total number of particles for each species is conserved after the chemical freeze-out such that for example $N_\pi = \text{const.}$. But from the temperature of the system, only the density of the particles, for example, n_π is known. However, both quantities can be related if an ideal fluid evolution is assumed, such that $N_\pi/S = n_\pi/s$, where S is the total entropy and s is the entropy density. With that, the conservation law $\partial_\mu (n_\pi u^\mu) = 0$ can be expressed for pions. The resonance decays break this conservation, however, new quantities for all particles can be constructed which are conserved again. For that, also the decays are taken into account. If for example the process $\rho \to \pi\pi$ is considered, the combination $n_\pi + 2n_\rho$ is still conserved. The system may then be described by equations of motion dependent on the temperature and the chemical potential μ. This is illustrated in fig. 3.3, which is meant as an example to describe the introduction of two freeze-out temperatures.

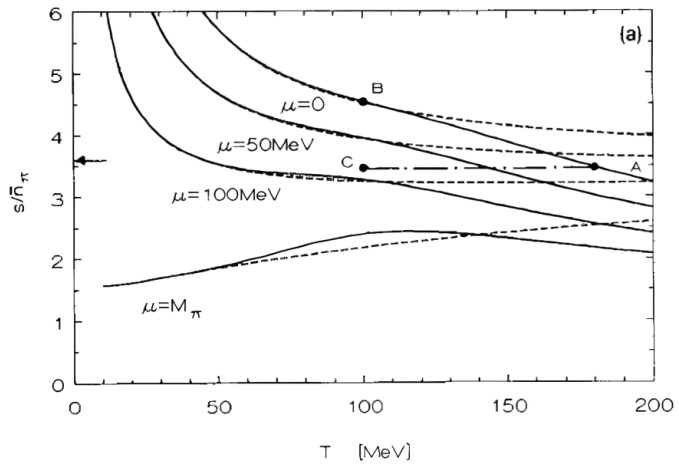


FIGURE 3.3: An example for the introduction of two freeze-out temperatures. The entropy per particle as a function of temperature. Taken from [36].

It shows the dependence of the ratio of the entropy over the conserved particle number n_π on the temperature and chemical potential. Suppose now $T_{\text{chem}} = 180\,\text{MeV}$ (point

A). Then the chemical potential at this temperature is zero due to the chemical equilibrium and the ratio $s/\bar{n}_\pi \simeq 3.5$. If the chemical equilibrium would be maintained for temperatures smaller than T_{chem}, the evolution would follow the curve at $\mu = 0$ and $s/\bar{n}_\pi$ would increase. However, it is assumed that $s/\bar{n}_\pi$ remains constant until the kinetic freeze-out, such that chemical potential builds up. In the depiction, this means that the evolution follows a horizontal line while the temperature decreases. If T_{kin} is reached, which is here at 100 MeV, a chemical potential of 86 MeV has built up. This shows, how the conservation of particle numbers drives the system out of the chemical equilibrium and how chemical and kinetic freeze-out can be differentiated in the model [36].

3.3.1 FastReso

FastReso [9] is an efficient method to calculate the final decay spectra of direct resonance decays directly from the hydrodynamic fields at freeze-out.
The usual way to model the particalization is to convert the hydrodynamic fields to particles by eq. 3.34 and then handle the resonance decays. The effect of the resonance decays on the final particle spectra can be mathematically treated by so-called decay maps $D_b^a(\mathbf{p}, \mathbf{q})$, which provide the Lorentz invariant probability of particle a with momentum q to decay to a particle b with momentum p. By summing all contributions from resonances decaying to a specific particle b, the final particle spectra can be obtained by [9]

$$E_{\mathbf{p}} \frac{dN_b}{d^3\mathbf{p}} = \sum_a \int \frac{d^3\mathbf{q}}{(2\pi)^3 2E_{\mathbf{q}}} D_b^a(\mathbf{p}, \mathbf{q}) E_{\mathbf{q}} \frac{dN_a}{d^3\mathbf{q}}. \tag{3.37}$$

The individual decay maps $D_b^a(\mathbf{p}, \mathbf{q})$ can be computed for each decay cascade by considering phase-space integrals, 4-momentum conservation and decay matrix elements. In contrast to this, in the FastReso method, the order of the integration in eqs. 3.34 and 3.37 are reversed, such that the decay maps are applied and then the integration over the freeze-out surface is performed. The formula for the final decay spectrum then becomes

$$E_{\mathbf{p}} \frac{dN_b}{d^3\mathbf{p}} = \frac{\nu_b}{(2\pi)^3} \int_\sigma g_b^\mu(p, u, T, \mu) d\sigma_\mu, \tag{3.38}$$

where the *vector distribution function* g^μ is defined as $g_a^\mu = f_a p^\mu$ for the primary resonances and is given by

$$g_b^\mu(p, u) \equiv \sum_a \frac{\nu_a}{\nu_b} \int \frac{d^3q}{(2\pi)^3 2E_{\mathbf{q}}} D_b^a\left(p^\nu q_\nu\right) f_a\left(-u^\sigma q_\sigma\right) q^\mu \tag{3.39}$$

for the decay products. The function $g_b^\mu(p, u, T, \mu)$ can be further simplified into a sum of a few irreducible Lorentz invariant weight functions and Lorentz vectors. These components only need to be computed once, the function $g_b^\mu(p, u, T, \mu)$ may then be constructed for all decay chains. This is of less effort compared to the standard procedure since the decays do not have to be treated on an event-by-event basis. The final particle spectra can then be computed by evaluating the Cooper-Frye integral for an arbitrary freeze-out surface. The reasoning of the FastReso method is explained in more detail in [9].

3.4 Comparing to experimental data

Employing all the previous steps, a relativistic heavy-ion collision can be modeled starting from the initial collision until the production of the final particles. Since the ultimate goal is to infer the properties of the individual stages with respect to the real physics in a collision, the model outcome has to be compared to experimental data. This can be done by considering different physical observables which focus on different aspects of the collision system. The only accessible observables are connected to the types, yields, momenta and spatial distributions of the produced particles since only the last stage of the free-streaming particles can be observed. Furthermore, the model output needs to be treated in the exact same way as the experimental data to allow for comparisons between them. This especially includes detector limitations that influence the physical observables. The most straightforward observables to consider are the averaged multiplicities for all particles. However, only single numbers for the produced particles may contain little information, therefore also the distributions of the multiplicities may be evaluated with respect to the rapidity or momentum. Other physical observables that are often used to describe particle production are the particle spectra. They are the yields of the specific particles in dependence on their momentum. With that also the kinematic properties of the particles are taken into account. The particle spectra are often given in dependence of the momentum transverse to the beam direction p_T. This is done because the momentum components longitudinal and transverse to the beam are independent of each other and p_T has lower uncertainties. Furthermore, the spectra are often built from particles observed only at midrapidity ($\eta < 0.5$), also because of the better detector coverage. The typical spectrum is given by $\frac{1}{N_\mathrm{ev}} \frac{1}{2\pi p_\mathrm{T}} \frac{d^2N}{dp_\mathrm{T}dy}$ in units of $[\mathrm{c}^2/\mathrm{GeV}^2]$, where N_ev is the number of events and y is the rapidity, which is given by $y = \frac{1}{2} \ln \frac{E+p_z}{E-p_z}$. The rapidity is approximately the same as the pseudorapidity if the mass of the particles is negligibly small compared to its momentum. Moreover, particle spectra are commonly produced for every centrality class individually based on the centrality classes from the experiment. The particle spectra provide already a lot more information than the multiplicities, however also the spatial distributions of the produced particles can be taken into account. For this, often the collective flow is considered, which describes the azimuthal distribution of the particles and reflects their collective motion. The flow is usually expressed in flow coefficients, which are the Fourier coefficients of the azimuthal distribution of the particles. Additionally, other observables like the mean p_T value or particle correlations may be used, however, most widely used for comparing hydrodynamic modeling to experiment are spectra and flow coefficients.

Chapter 4

Methods

The work presented in this thesis employs methods from the field of machine learning (ML), where the algorithms are not explicitly programmed to make predictions or decisions but are rather learning underlying rules from data to do so. For this, the algorithms are trained using *training* data to make accurate predictions. These are verified on *test* data, which is data that the model has not seen during the training process. This procedure ensures that the model captures the underlying data dependencies instead of just memorizing the training samples and therefore *generalizes* well. ML comprises a wide variety of different methods that are closely related to computational statistics and that are applied in diverse fields like medicine, computer vision or speech recognition. The various ML methods can be assigned to different approaches, of which the two most important ones are supervised and unsupervised learning. In supervised learning, the goal is to find a function that infers outputs based on inputs. The function is constructed by regarding training samples of input-output pairs. Mathematically, this can be expressed by finding a function f_θ with model parameters θ that maps the inputs x from an input space X to the target y from output space Y

$$y = f_\theta(x). \tag{4.1}$$

The form of x and y can be different depending on the problem. x may be a $n \times 1$ column vector with entries for each of the n data points, but it may also be of the form $n \times m$, with m being the number of input variables. Similarly, y may have different forms, it could be a scalar value, a vector of output variables or even a matrix. Based on the type of the target variable, two cases can be distinguished. If the output variable is of categorical nature, so that it can only take on a certain finite number of values, the method is called classification, whereas if the output variable can take on continuous numerical values, it is called regression. For example, classification is used to categorize images or to make decisions whereas regression is used for interpolation or extrapolation of data. In unsupervised learning, where no target variables are available, the goal is primarily to find structures and similarities in the data. Examples for this are density estimation, where a probability density distribution is inferred from the data, or cluster analysis, where similarities between data points are identified and assigned to clusters. In the following, the ML methods that will be used in this thesis will be introduced. Why and how they are used will be explained in the analysis part of this thesis 5.

4.1 Neural networks

An artificial neural network (ANN), also often abbreviated only with neural network (NN) is a computational learning system that is inspired by the functioning of the brain. It is an algorithm in the field of supervised learning, which can be used for classification as well as for regression tasks. NNs are able to find and model complex relationships between inputs and outputs and are therefore often used in data analysis. A NN is composed of a network of simple processing units, the artificial neurons or nodes, which are inspired by biological neurons. These artificial neurons can be mathematically described by

$$y = \phi(z) = \phi\left(\mathbf{w}^T\mathbf{x} + b\right). \tag{4.2}$$

They transform an input vector $\mathbf{x}(\mathbf{x} \in \mathbb{R}^m)$ by weighting it with weights $\mathbf{w}(\mathbf{w} \in \mathbb{R}^m)$ and adding a scalar bias term b. The outcome is then fed into an *activation function* ϕ and the result is the output for the neurons. Fig. 4.1 A and fig. 4.1 B show two representations of such a neuron, A with all elements and B in a simplified version.

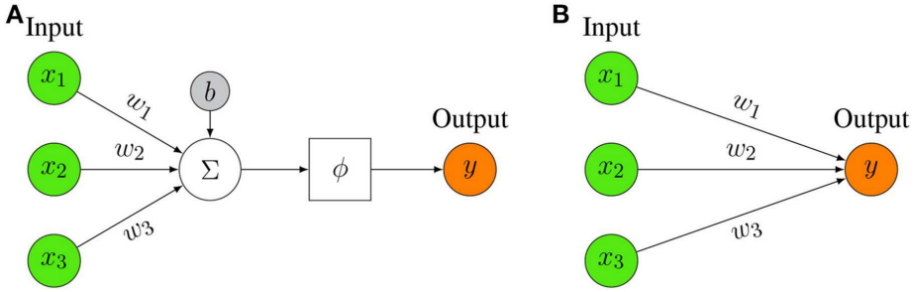


FIGURE 4.1: A) Representation of a neuron with inputs x_i, weights w_i, bias b and activation ϕ. B) Simplified representation of neuron that is commonly used. Taken from [37].

The choice of the activation function is important for the behavior of the neuron model. Using the identity function for example would result in simple linear regression or a linear discriminant function for classification. However, to model non-linear problems, the activation function is usually chosen to be non-linear as well. Popular examples for such functions are the sigmoid function, the hyperbolic tangent or the Rectified Linear Unit (ReLU), which is widely used in the field of deep learning.
For the modeling of complex non-linear problems, multiple simple neurons can be connected to form a neural network. The connections are realized by using the outputs of some neurons as inputs to other ones. Depending on how the neurons are connected, different arrangements (architectures) of the neurons are possible, which are suitable for different types of problems. Here, only the simple case of a feed-forward neural network will be described, information about other architectures may be taken from [38]. The feed-forward NN (FFNN), as the name suggests, only consists of neurons connected in the forward direction, meaning that information is strictly transported from the inputs to the outputs without any cycles. The network is usually organized in layers of neurons as illustrated in fig. 4.2.
The first layer is processing inputs and therefore it is referred to as the input layer. The last layer is accordingly called the output layer and any additional layers that might occur in between are so-called hidden layers. Mathematically the network can then be described by subsequent execution of the single neuron transformation in eq. 4.2 for all neurons in the layers:

$$f(\mathbf{x}) = \varphi^{(2)}\left(W^{(2)}\varphi^{(1)}\left(W^{(1)}\mathbf{x} + \mathbf{b}^{(1)}\right) + \mathbf{b}^{(2)}\right). \tag{4.3}$$

The weights of the individual neurons are now encoded layer-wise in the weight matrices $W^{(l)}$, which are matrices of $k^{(l)} \times k^{(l-1)}$, where $k^{(l)}$ is the number of neurons in the lth layer and $k^{(0)}$ is the dimensionality of the input x. The bias becomes a $k^{(l)}$-dimensional vector $\mathbf{b}^{(l)}$, which collects all biases of the neurons in layer l. The scalar activation function is now applied element-wise to each output of the neurons in a layer such that a vector of length $k^{(l)}$ is returned. The output is then obtained by treating the output of the first layer as the input of the second layer and so on. By adding more layers or nodes to the network, more complex non-linear functions can be modeled. In fact, it

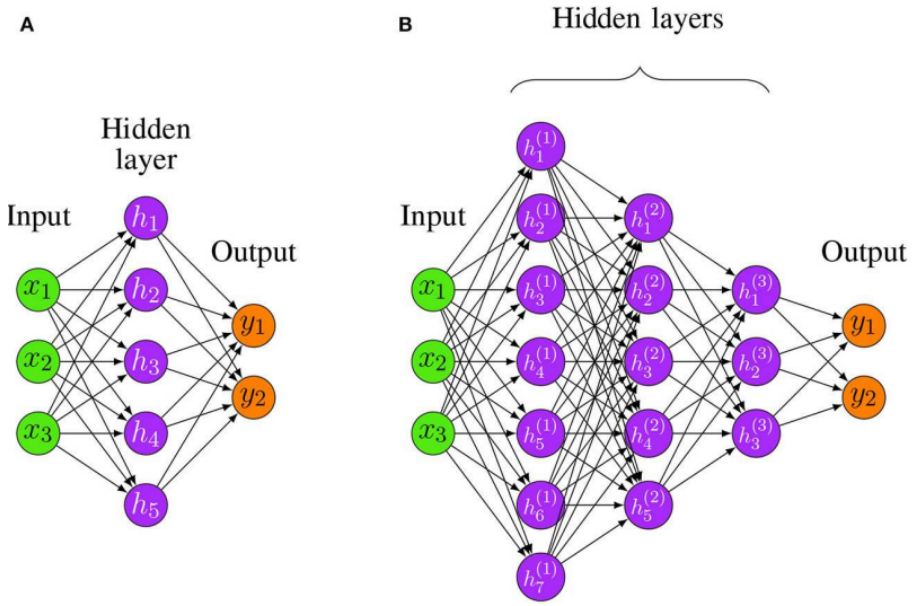


FIGURE 4.2: A) A shallow FFNN with one hidden layer and one input and one output layer. B) A deep FFNN with three hidden layers. Taken from [38].

was postulated that a NN with one hidden layer and a sufficient number of neurons is already capable of approximating *any* continuous function by adjusting its weights and bias values. This is known as the universal approximation theorem and it is the reason why NNs are so widely used. However, the theorem does not provide information on *how* the network has to be trained to achieve this. In practice it is very difficult to construct such a network because its width can get exponentially large. A solution to this was found by introducing additional hidden layers to the network, which gives rise to the so-called deep neural networks (DNNs). For deep neural networks with a limited number of hidden units the universal approximation theorem has been shown to remain valid, and for these networks efficient learning algorithms have been found. For this reason, they are used in practice rather than one-layered neural networks.

4.1.1 Neural network training

So how can the neural network weight parameters $\mathbf{w}$ be adjusted to solve specific tasks? In the supervised setting, they can be optimized using the training data. For this, a loss function has to be defined, which quantifies the performance of the network. An example for this is the mean squared error for regression (MSE), which is defined as

$$Q(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} Q_i(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} \left(f(x_i, \mathbf{w}) - y_i\right)^2, \tag{4.4}$$

where y_i is the target value for the input x_i, $f(x_i, \mathbf{w})$ is the prediction of the NN for the input x_i, n is the number of training samples and $\mathbf{w}$ are the weight parameters. If the network is able to fully reproduce the data generating process, the loss function is zero, the more they differ, the larger is the loss. Therefore, to approximate the data generating process, the loss function has to be minimized with respect to the weight parameters. This can be done by using a gradient descent algorithm, of which the simplest form would give rise to the iteration rule

$$\mathbf{w}_{j+1} = \mathbf{w}_j - \eta \nabla Q(\mathbf{w}) = \mathbf{w}_j - \frac{\eta}{n} \sum_{i=1}^{n} \nabla Q_i(\mathbf{w}), \tag{4.5}$$

where η is the gradient descent step size of the algorithm, also known as *learning rate*, since it defines how fast a network can learn. For large training datasets, the calculation of all gradients in eq. 4.5 is inefficient, therefore in practice only a part of the gradients is used to approximate the true gradient of $Q(\mathbf{w})$. This gives rise to the stochastic gradient descent algorithm, where the parameters are updated using only single training samples or mini-batches of training samples:

$$\mathbf{w}_{j+1} = \mathbf{w}_j - \eta \nabla Q_i(\mathbf{w}). \tag{4.6}$$

The training samples that are used in each training iteration are only a subset of the full training set and are sampled randomly from it.
The calculation of the gradients in eq. 4.6 is taken care of by another algorithm, the backpropagation algorithm, which is highly efficient. Backpropagation computes the gradient of the loss function with respect to each weight parameter by the chain rule. The algorithm iterates backward through the network while computing the gradients layerwise and therefore omits redundant calculations that would occur if the gradient is computed for each parameter individually. A more detailed explanation can be found in [39]. Stochastic gradient descent together with backpropagation provides an efficient method to train the parameters of a NN. In practice, variants of these methods are combined into an optimization algorithm, which performs the optimization steps computationally.

4.1.2 Hyperparameters

There are a lot of parameters, which define the architecture as well as the training process of a NN. Since these parameters are set before the training process and are not optimized in it they are called hyperparameters. The hyperparameters have a large impact on the performance and training process of the NN and thus it is crucial to find an optimal set of hyperparameters for the NN to solve the task in the best possible way. However, the optimal parameters are strongly dependent on the specific problem, no universally applicable values for them exist. This is why the hyperparameters are often optimized themselves using strategies like grid search or Bayesian optimization. In the following, important hyperparameters of NNs together with their impact on the training will be elaborated shortly. Some of the hyperparameters like the activation function or learning rate were already introduced, here also new ones will be discussed.

- **Number of layers and number of nodes per layer** The number of hidden layers and the number of nodes per layer have a large impact on the performance of a NN. If the number of layers and nodes is too small, the produced NN model may be too simple to capture the complexity of the data, thus missing important information about the relationship between inputs and outputs. This behavior is referred to as *underfitting*. On the other side, if the number of layers and nodes of the NN is too large, its complexity may exceed the one of the data, such that even the data noise is reproduced by the model, a behavior that is called *overfitting*. Therefore, an appropriate size of the NN has to be found to ensure the best performance. In practice, NNs are often chosen to be rather large with multiple hidden layers and nodes per layer such that underfitting is prevented. Overfitting is then taken care of by *regularization* techniques [40] which lower the complexity of NNs by adding a penalty term to the loss function that penalizes

large numbers of nodes. However, the disadvantage to such large NNs is that the computational cost increases with the number of nodes because of the gradient calculations. Thus, parallelization techniques commonly need to be applied for the training of complex NNs.

- **Activation function** The activation function usually introduces the non-linearity into the NN model. There are many choices for its functional form depending on the problem. One example is the Sigmoid function ($\Phi(z) = 1/(1+e^{-z})$), which is often used for classification problems since it returns only values between 0 and 1, which can be associated with probabilities. Another often used choice is the hyperbolic tangent ($\Phi(z) = \tanh(z)$), which maps values to the interval [-1, 1] and has a similar shape as the Sigmoid. It has the advantage that negative z values are also mapped to negative output values and that z values around zero are also mapped to the region at zero. The Sigmoid function and the hyperbolic tangent are both differentiable and monotonic activation functions which are important properties for calculating the gradients. However, the most used activation function today is the ReLU ($\Phi(z) = \max(0, z)$), which is monotonic but not differentiable at 0. It is so widely used because its evaluation, as well as the gradient calculation, is computationally cheap: this is especially of advantage for the training of large networks in the context of computer vision.

- **Weight initialization** Neural networks are optimized by applying the gradient descent algorithm on the loss function and progressively updating weights. For this process, the optimization algorithm requires a starting point in the weight space to begin the optimization. The procedure to select the starting point is called *weight initialization*. The weight initialization largely impacts the training process, since the path to and the position of the local minimum in the optimization process depend on it. A bad initialization can result in a very slow or not converging optimization. Usually, the weights are initialized randomly according to a distribution around zero, which can be for example a uniform or a Gaussian distribution. However, also more advanced methods like Xavier initialization [41] have been developed, that take into account the chosen activation function to achieve a faster training process.

- **Number of epochs** In each epoch, the network is trained on all training samples. Therefore, the number of epochs defines how often the network is trained on the full training set. For small numbers of epochs, the training may not converge, whereas for a large number the network may overfit, therefore a reasonable value has to be found. In practice, the optimal value can be identified by finding the minimum in the loss function for the test dataset, and if the NN starts to overfit, this value will increase again. Since the test loss is monitored anyway during training, the number of epochs can be optimized without any further effort.

- **Batch size** The batch size is the number of training samples that are fed through the network before an optimization step is done. Choosing a small batch size leads to a faster training process and less memory usage because fewer gradients have to be calculated for each step. However, the estimate for the gradient will be less accurate than for larger batch sizes since only a subset of gradients is considered.

- **Learning rate** The learning rate defines the step size for the gradient descent algorithm. It is one of the most important hyperparameters because of its large impact on the convergence of the training process. For too small values, the algorithm would converge too slowly or get stuck in a local minimum, whereas for

too large values it may overshoot in an unstable training process such that the minimum is missed. Furthermore, the optimal value usually decreases during training, and this makes it even harder to optimize it. Hence, the learning rate is often varied during training according to a predefined schedule or in an adaptive way to optimize convergence and its starting value is optimized externally.

- **Loss function** The loss function quantifies the performance of the network. It is chosen according to the type of the problem. For regression tasks, the Mean Squared Error function, which was already introduced in equation 4.4, is usually used. For classification tasks, the Cross-Entropy loss function is utilized, which takes into account the categorical nature of the target variables. However, there are in principle many more loss functions that could be used and its selection is to a large degree the choice of the user.

- **Optimization algorithm** The optimization algorithm is the heart of the training process of a NN. It is the implementation of the stochastic gradient descent and backpropagation algorithm and ultimately performs the computational optimization steps. There are many different variants of which the most used methods are simple stochastic gradient descent (SGD), Adagrad [42] or Adam [43]. Depending on the algorithm, the rate of convergence can be quite different, such that a good choice can lower the computational effort drastically. To find the best performing optimizer, the different methods can be tested and evaluated, however, Adam is nearly always a good choice.

4.2 Uncertainty quantification for neural networks

In recent years, neural networks have been applied in many areas of science and technology. With new and more critical tasks, it has become increasingly important that the networks not only have high precision but also provide information about the confidence of their prediction. However, simple NNs only give point predictions without any measure of their uncertainty or confidence. This is especially critical in scientific applications, where uncertainties are crucial parts of any analysis. Therefore, to solve this problem, several methods have been developed in the past that can estimate the uncertainty within the context of NNs [44]. One of these will be presented in the following, but at first, different types and sources of uncertainty in the prediction will be discussed.
Examples of uncertainties arising in the process of fitting a model like a NN to data are errors from the variability of real-world situations, errors inherent to the measurement system, errors in the architecture of the model, errors in the training procedure, or errors caused by unknown data [44]. However, the different sources can be separated into two general types of uncertainty, the *data uncertainty* (also *aleatoric* or *statistical* uncertainty) and the *model uncertainty* (also *epistemic* or *systematic* uncertainty). As the name suggests, the data uncertainty accounts for all errors inherent to the data itself like the measurement process or statistical fluctuations. This uncertainty is irreducible and will be always apparent if real-world data is used even if more data is available. In contrast to this, the model uncertainty is related to modeling errors like insufficient model complexity, errors in the training process or missing information due to unknown data. Theoretically, this type of uncertainty is reducible by adjusting the model or training process or by providing more data to the model, but it is not always possible to remove it completely.

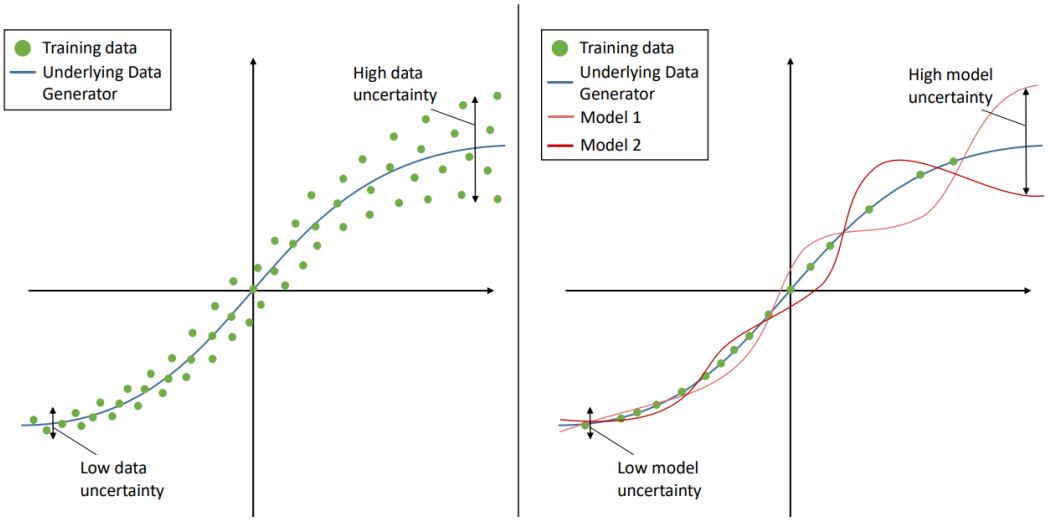


FIGURE 4.3: Depiction of data and model uncertainty. Left: Low and high data uncertainty, visible by the spread of the data. Right: Low and high model uncertainty, visible by the spread of appropriate models. Taken from [44].

The two types of uncertainty are depicted in fig. 4.3. On the left side low and high data uncertainties are illustrated which manifests in different spreads of the data distribution. The model uncertainty is shown on the right side: it can be depicted by the variation between a number of reasonable models that describe the data equally well. The two shown models for example may both be suited to describe the underlying data generating process, however, they differ substantially in regions where the model is poorly constrained because of limited data. This is therefore connected to larger model uncertainty. This becomes especially apparent in regions where there is no data available, so-called out-of-distribution regions, such that the model is not constrained at all and the model uncertainty increases.

4.2.1 Uncertainty quantification via ensemble methods

To quantify uncertainties in the context of deep learning, various methods are available. An overview can be found at [44], on which this section is based. Here, only ensemble methods will be discussed in detail as they will be used later.
The idea of ensemble methods is to obtain the prediction not on the basis of only one model, but to combine several different models in an ensemble model. Because the individual members compensate for their respective weaknesses, the ensemble model is superior to the individual ones and the overall predictive uncertainty decreases [45]. There are different ways to combine the predictions f_i of the members $i \in 1, 2, ..., M$ of the ensemble, of which the simplest one is to just average them by

$$f(x) := \frac{1}{M} \sum_{i=1}^{M} f_i(x), \tag{4.7}$$

where x is the model input. Already this intuitive approach of ensembling has been applied successfully for example in bioinformatics or climate modeling [44]. However, there are also more advanced combination methods for example weighting and averaging the predictions of the members.
Additionally to the reduced predictive uncertainty, an ensemble approach can be used to infer an estimate for the model uncertainty, which is the main reason why they will

be used in the analysis presented in this work. For this, the spread of the different ensemble members can be taken into account. Mathematically, this is often quantified by the standard deviation of the member predictions given by

$$s_M = \sqrt{\frac{1}{M}\sum_{i=1}^{M}(f_i - \bar{f})^2}, \tag{4.8}$$

such that a model like in 4.4 arises. However, it becomes apparent from equations 4.7 and 4.8, that a meaningful estimate for the model uncertainty and a better generalization of the ensemble can only be achieved if two requirements are fulfilled:

- the individual performance of each member has to be as optimized as possible
- the individual members have to be as diverse as possible

Diversity needs to be introduced to decrease the correlation between the errors of the members. If the correlation is large, the models will all tend to deviate in the same direction, such that the estimate for the model uncertainties is incorrect. To introduce diversity to the models, there are different methods in the field of NNs. The first option is to use random weight initialization and random data shuffling in the training process. With this, each network is trained starting from a different point in the loss landscape and therefore also results in a different local minimum. Data shuffling has a similar effect since the network is trained in batches, its path varies during training, and this leads to different local minima in the vicinity of the global minimum. Another option to introduce variety into NNs is to use Bagging (Bootstrap aggregating) or Boosting, which aim at varying the training data for ensemble members. Bagging means, that each NN of the ensemble is trained on a subset of the whole training data that is sampled from it uniformly with replacement. This alters the training between NNs and results in different local minima, thus varying the prediction of the models. Boosting refers to a training procedure where the networks are trained one after another and the probability of sampling a sample for the subsequent training set is based on the outcome of the already trained ensemble. In this way, training samples for which the prediction is poor get a larger weight, so the next network improves most where the ensemble performs worst. This results in a better generalization of the ensemble. As another option to maximize diversity among the ensemble members, different network architectures for the NNs in the ensemble or data augmentation can be used, where the input data is augmented randomly. It has been shown, that random initialization and random shuffling are already sufficient to induce diversity into the ensemble and that bagging may even lead to worse performance because the individual NNs are not trained on the whole data set [46].
Because these strategies are already standard elements of any neural network, all that is required to create an ensemble of NN is to train multiple NNs and combine their predictions. This is easy to implement but leads to significant computational overhead. For each additional member in the ensemble, the memory consumption and computational effort of training and testing increase linearly, which is often the limiting factor for the usage of ensembles in practice. However, because the ensemble members are completely independent of each other, the training process, as well as the evaluation, is fully parallelizable, thus decreasing the time consumption of these steps.

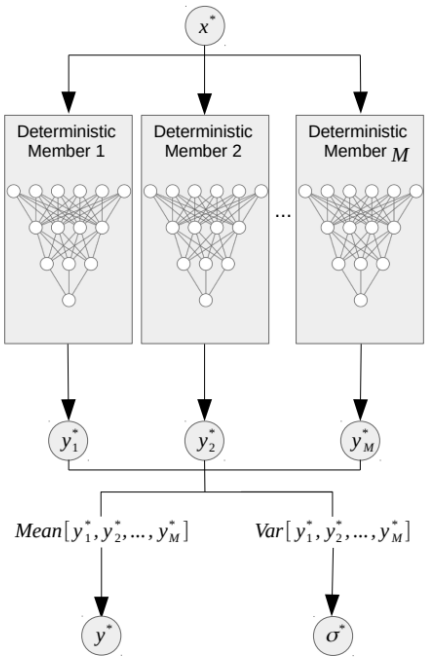


FIGURE 4.4: Visualization of an ensemble method. The individual predictions of the ensemble members are combined to a mean prediction and uncertainty estimate. Taken from [44].

4.3 Markov chain Monte Carlo

In the course of the analysis in this thesis, Bayesian inference will be used to obtain probability densities of model parameters. For that, the posterior density will be inferred from a probabilistic model and high dimensional integrals will be performed to get marginal probability densities for each parameter. Such a procedure is typical for Bayesian inference, however, the calculation of the densities is usually intractable for all but the simplest models. Also, in this case, an analytic treatment will not be possible, because the probabilistic model will not have an analytic expression. To nonetheless compute the considered quantities, approximating numerical methods have to be employed.

The standard method to obtain probability distributions or to integrate numerically in high-dimensional spaces is Monte-Carlo (MC) sampling or as it is also called Monte-Carlo integration. The idea is to sample independently and randomly from a probability distribution to approximate the desired quantity. For example, instead of calculating the mean of a normal distribution analytically, it can be approximated by computing the sample mean of a number of random samples drawn from a normal distribution. The error of this approximation decreases with $1/\sqrt{n}$, with n being the number of drawn samples. This method can also be used to infer the probability density numerically. For that, random samples are drawn from the density and for a sufficiently large number of them, the distribution of samples will converge to the true density. Nevertheless, simple Monte Carlo sampling is inefficient because of the random sampling of the probability densities, which is often done by the inefficient rejection method. This is particularly critical in high-dimensional spaces, where the number of samples needs to increase exponentially with respect to the dimensions to ensure a sufficient population of the probability space. To overcome this problem, Markov chain Monte Carlo

(MCMC) methods have been developed. In these methods, samples are drawn randomly but not independently by constructing a so-called Markov chain. Each element of the chain is sampled in dependence on its preceding element and *only* its preceding element. This property is known as the Markov property. It can be mathematically expressed in terms of conditional probabilities. Assuming a sequence of samples $z_1, z_2, ..., z_n$ from a sequence of random variables $Z_1, Z_2, ..., Z_n$, the Markov property is fulfilled if [40]

$$q(Z_{n+1} = z|Z_1 = z_1, Z_2 = z_2, ..., Z_n = z_n) = q(Z_{n+1} = z|Z_n = z_n), \tag{4.9}$$

where $q(Z_{n+1} = z|Z_n = z_n)$ or in short $q(z|z_n)$ is the probability of moving from state z_n to state z. Given the probability $q(z|z_n)$, a Markov chain may then be constructed by defining a starting point and generating new elements z by drawing samples from $q(z|z_n)$. By this procedure, the model performs a random walk based on the probability $q(z|z_n)$, and if set correctly, it will converge to the desired quantity for a sufficiently long chain. However, the next state probability distribution $q(z|z_n)$ has to be inferred from the target probability distribution $p(x)$ that should be integrated over. This is not always possible such that more general algorithms have to be used which split the dependence on the predecessor into a part of generating a proposal and a part of accepting the proposal. One example of this is the Metropolis algorithm. Within this algorithm, a proposal z' for the next element in the Markov chain is generated from a proposal probability distribution $q(z'|z)$ which is taken to be symmetric $q(z'|z) = q(z|z')$. An example of this would be a Gaussian distribution. Then, in a second step, the proposal is accepted or dismissed according to the Metropolis criterion [40]

$$r(z', z) = \min\left(1, \frac{p(z')}{p(z)}\right), \tag{4.10}$$

where $r(z', z)$ is the acceptance probability ranging from 0 to 1 and $p(z')$ and $p(z)$ are the target probabilities for states z' and z. The proposed new state is then accepted with probability r. In practice, this means that a random number u is sampled from a uniform distribution, and the proposal is accepted if $u < r$ and dismissed otherwise. If accepted, the proposal is added as the new element to the chain, otherwise, the predecessor is added again. By accepting the steps according to the target probability $p(x)$, the chain will quickly converge to the region of the highest probability, such that the equilibrium distribution can be computed efficiently. The convergence is dependent on the starting point of the chain, if this is chosen far away from the equilibrium distribution, a large number of steps is needed to reach it. This may distort the resulting distribution, to prevent it often the first samples from a chain are excluded, the so-called burn-in samples. Besides the above introduced Metropolis algorithm also other algorithms in the field of MCMC exist, which use different proposal probability distributions and acceptance criteria to infer the equilibrium probability distribution. Some examples can be found in the [47]–[49].

To ensure, that the Markov chain has converged sufficiently, the sampling error can be considered. For MC methods, this error decreases with the known $1/\sqrt{N}$ dependence, where N is the number of samples. For MCMC, however, the individual elements of the chain are not independent, such that sampling error actually decreases by $\sqrt{\tau_f / N}$ [50], where τ_f is the integrated autocorrelation time. τ_f / N can therefore be seen as the *effective* number of samples, where τ_f is a measure of how far apart two samples in the chain have to be for considering them to be independent of each other. The integrated autocorrelation time is given by [50]

$$\tau_f = \sum_{\tau=-\infty}^{\infty} \rho_f(\tau), \tag{4.11}$$

where $\rho_f(\tau)$ is the normalized autocorrelation function of the stochastic process that generated the chain of f. $\rho_f(\tau)$ can be estimated from a finite chain $\{f_n\}_{n=1}^{N}$ by [50]

$$\hat{\rho}_f(\tau) = \hat{c}_f(\tau)/\hat{c}_f(0), \tag{4.12}$$

where

$$\hat{c}_f(\tau) = \frac{1}{N-\tau} \sum_{n=1}^{N-\tau} \left(f_n - \mu_f\right)\left(f_{n+\tau} - \mu_f\right) \tag{4.13}$$

and

$$\mu_f = \frac{1}{N} \sum_{n=1}^{N} f_n. \tag{4.14}$$

For a finite chain, evaluating eq. 4.11 is not possible but has to be approximated. This is usually done by adding up all terms up to some limit $M \ll N$ [50]:

$$\hat{\tau}_f(M) = 1 + 2 \sum_{\tau=1}^{M} \hat{\rho}_f(\tau). \tag{4.15}$$

M is chosen much smaller than N which reduces the variance but adds a bias. Sokal [50] suggests using the smallest M for which $M \geq C\hat{\tau}_f(M)$ with $C = 5$.
With the autocorrelation time, the sampling error of the MCMC simulation can be controlled. To achieve, for example, an accuracy of 1 %, the chain has to consist of $N = 10000\tau_f$ elements. A more detailed discussion about the convergence of MCMC simulations can be found in [50].

Chapter 5

Analysis

With the introduction to the physics of heavy-ion collisions and their modeling in the previous chapters, the basis has been laid to accomplish the goal of this work, the quantitative estimation of central characteristics of the evolution of heavy-ion collisions. As already explained in chapter 3, these characteristics are not accessible via any direct measurement but have to be inferred indirectly from physical observables. For that, the desired characteristics can be treated as free parameters of a model simulating heavy-ion collisions, which is then optimized to reproduce the experimentally observed data. In the following, a new approach to this is discussed and applied. At first, the choice of the model, physical observables, and experimental data is described in sec. 5.1, then the general procedure for the optimization is laid out in sec. 5.3, and ultimately the optimization is performed in sec. 5.4, 5.5 and 5.6.

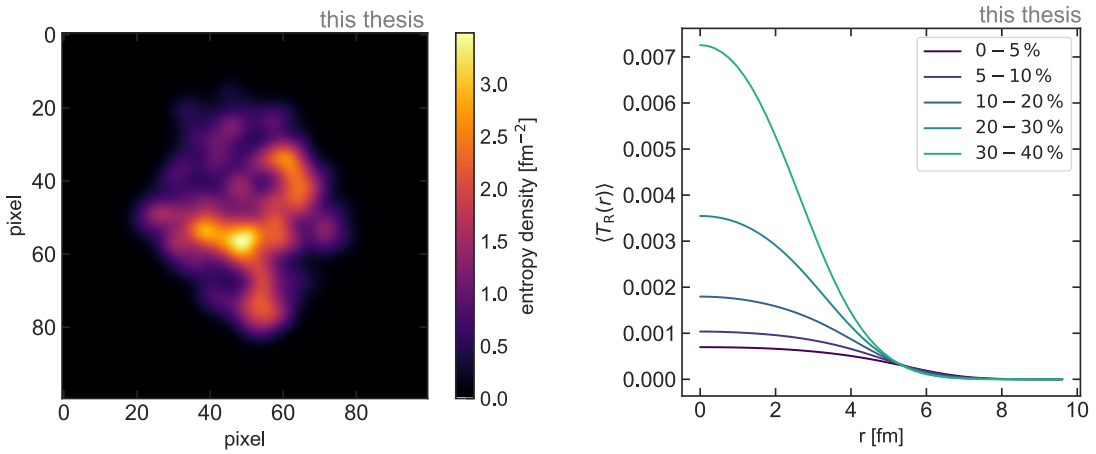
5.1 Analysis setup

As the first step in this analysis, the theoretical model, as well as the experimental data used as a reference, have to be defined. This is an important step because meaningful conclusions about the real physical properties in heavy-ion collisions can only be obtained from the parameters if the model describes the evolution of the system realistically. In this thesis, the strongly interacting matter evolution is described by combining the models introduced in chapter 3. The initial state of the heavy-ion collision is described by Trento, the evolution of the QGP by Fluidum and the freeze-out and resonance decays by FastReso. This setup is based on [24], the predecessor of this work, with some changes. For the optimization of the parameters, a completely new framework is developed. In the following, the settings for the individual sub-models of the simulation are laid out.

Initial conditions

To generate initial conditions, Trento is used as it was introduced in section 3.1. The parameters are chosen in accordance with ref. [8] similar to ref. [24]. The reduced thickness p is fixed to $p = 0$, the fluctuation parameter to $k = 1.4$, the nucleon width to $w = 0.6\,\text{fm}$ and the inelastic nucleon-nucleon cross section is set to $\sigma_{\text{inel}}^{\text{NN}} = 6.18\,\text{fm}^2$. Using this set of parameters, the initial entropy densities for the fluid evolution are obtained with the following procedure:

1. Generate transverse entropy densities $T_{\text{R}}(x, y)$ with Trento for 10^5 events.
2. Compute centrality classes 0-5%, 5-10%, 10-20%, 20-30%, and 30-40% using the event pseudo-multiplicities and eq. 3.1 and sort events into these centrality classes.
3. For each centrality class, calculate an averaged transverse entropy density $\langle T_{\text{R}}(r) \rangle$. For that, center each entropy density $T_{\text{R}}(x, y)$ to the center-of-mass and rotate it randomly before averaging the profiles to $\langle T_{\text{R}}(r) \rangle$ on the x-axis.



(A) Example of a transverse entropy density $T_R(x, y)$ generated by Trento. One pixel corresponds to 0.1 fm.

(B) Averaged transverse entropy densities $\langle T_R(r) \rangle$ for different centrality classes. Computed with settings as given in tab. 5.1.

FIGURE 5.1: Trento entropy density computation.

4. Scale the entropy densities $\langle T_R(r) \rangle$ with a normalization of $\frac{Norm}{\tau_0}$

$$s(r) = \frac{Norm}{\tau_0} \langle T_R(r) \rangle, \tag{5.1}$$

to obtain the initial condition $s(r)$ for the fluid dynamic evolution.

In contrast to event-by-event simulations, here the individual entropy densities are not taken directly as the input to the fluid evolution model but are rather averaged to obtain the mean entropy densities as initial conditions. This is done because Fluidum considers only the mean evolution of the system. For the sample average to be an accurate representation of the mean, the number of generated events has to be large enough. The value of 10^5 is based on ref. [24], but this will be verified in this analysis once more. In step 3, the entropy densities are reduced from the two-dimensional form $T_R(x, y)$ to the one-dimensional $\langle T_R(r) \rangle$. This is valid because the average entropy densities are azimuthally symmetric for a large number of events.
In the last step, the computed entropy densities for all centrality classes are scaled by a factor of $Norm/\tau_0$. This scaling factor is introduced as a free parameter of the model, $Norm$ describes here a normalization factor, and τ_0 is the thermalization time. The thermalization time is taken out here to decouple the Bjorken flow at early times from the normalization.
The setup for the initial conditions is summarized in table 5.1. In fig. 5.1A and 5.1B, an example of one transverse entropy density $T_R(x, y)$ is shown together with the computed average transverse entropy densities $\langle T_R(r) \rangle$ for different centrality classes. The treatment of the initial conditions described here is identical to ref. [24], more information can be found there.

Fluid dynamic evolution of the QGP

Following the description of the evolution of heavy-ion collisions in chapter 2.3, the next stage after the initial collision and before the fluid evolution model is the pre-equilibrium phase. This will not be modeled for this analysis, such that the initial velocity at the beginning of the fluid evolution is assumed to be zero. Thus, the generated entropy densities are used directly as initial conditions for the fluid dynamic evolution.

The evolution of the QGP is described by relativistic dissipative fluid dynamics, and the equations of motion arising from this approach (as given in 3.2.2) are solved by the software package Fluidum [7]. In this analysis, the azimuthally averaged transverse momentum spectra at mid-rapidity are used to compare to data (more on that in sec. 5.1). For this reason, only the azimuthally symmetric background part $\mathbf{\Phi}_0(\tau, r)$ (as introduced in sec. 3.2.4) will be considered in the model. Any azimuthally and rapidity-dependent perturbations from the fluctuation part $\mathbf{\Phi}_1(\tau, r, \phi, \eta)$ are neglected. The used implementation of Fluidum features shear and bulk viscous dissipation and a state-of-the-art thermodynamic equation of state [7]. For the analysis, it is assumed that the ratio of the shear viscosity over entropy η/s is independent of the temperature. Additionally, for the bulk viscosity over entropy ratio ζ/s a dependence on the temperature is set, which is assumed to be given by a Lorentzian function [24]

$$\zeta/s = \frac{(\zeta/s)_{\max}}{1 + \left(\frac{T - T_{\text{peak}}}{\Delta T}\right)^2}, \tag{5.2}$$

with the peak temperature $T_{\text{peak}} = 175\,\text{MeV}$ and $\Delta T = 24\,\text{MeV}$ [51]. Furthermore, within Fluidum, the shear and bulk relaxation times are assumed to be given by the equations

$$\frac{\tau_{\text{shear}}}{\eta/(\epsilon + p)} = 5, \qquad \frac{\tau_{\text{bulk}}}{\zeta/(\epsilon + p)} = \frac{1}{15(\frac{1}{3} - c_s^2)^2} + \frac{a}{\zeta/(\epsilon + p)}, \tag{5.3}$$

where ϵ is the energy density, p the pressure, $c_s(T)$ the speed of sound and $a = 0.1\,\text{fm/c}$ is a small offset to ensure a causal evolution of the radial expansion [24].
For this analysis, only the ratio of shear viscosity to entropy η/s and the maximum bulk viscosity $(\zeta/s)_{\max}$ are free parameters that will be optimized. They are chosen because they have the largest impact on the evolution of the fluid. In table 5.1, the setup for the fluid evolution model is summarized.

Hadronization and resonance decays

To model the hadronization and resonance decays, the software package FastReso is used as it was introduced in section 3.3. In addition to pre-computing the decay maps, also the freeze-out integrals over space-time rapidity and azimuthal angle may be pre-computed for the present study. This is possible because only the azimuthally symmetric background part of the fluid evolution is considered. With that, the freeze-out surface can be reduced to $1+1$ dimensions described by $(\tau(\alpha), r(\alpha))$, where $\alpha \in (0, 1)$ parametrizes the position on the surface. The integral over the freeze-out surface reduces to [24]

$$\begin{aligned}\frac{dN}{2\pi p_{\text{T}} dp_{\text{T}} dy} =& \frac{\nu}{(2\pi)^3} \int_0^1 d\alpha \tau(\alpha) r(\alpha) \\ &\times \left\{ \frac{\partial r}{\partial \alpha} \left[K_1^{\text{eq}} + \frac{\pi_\eta^\eta}{2(\epsilon + p)T^2} K_1^{\text{shear}} + \frac{\pi_\Phi^\Phi}{2(\epsilon + p)T^2} K_3^{\text{shear}} - \frac{\pi_{\text{bulk}}}{\zeta/\tau_{\text{bulk}}} K_1^{\text{bulk}} \right] \right. \\ &\left. - \frac{\partial \tau}{\partial \alpha} \left[K_2^{\text{eq}} + \frac{\pi_\eta^\eta}{2(\epsilon + p)T^2} K_2^{\text{shear}} + \frac{\pi_\Phi^\Phi}{2(\epsilon + p)T^2} K_4^{\text{shear}} - \frac{\pi_{\text{bulk}}}{\zeta/\tau_{\text{bulk}}} K_2^{\text{bulk}} \right] \right\},\end{aligned} \tag{5.4}$$

where $K_i^{\text{eq}}(p_{\text{T}}, u^r)$, $K_i^{\text{shear}}(p_{\text{T}}, u^r)$ and $K_i^{\text{bulk}}(p_{\text{T}}, u^r)$ are kernels integrated over the azimuthal angle and the rapidity.

initial conditions	fluid evolution	hadronization and decays
model: Trento [8]	model: Fluidum [7]	model: FastReso [9]
$p = 0$	relativistic, dissipative fluid	$\sim$ 700 considered decays
$k = 1.4$	background field $\mathbf{\Phi}_0(\tau, r)$	$<$ 3 GeV mass resonances
$w = 0.6$	$(\eta/s(T)) = \text{const}$	partial chem. equilibrium
$\sigma_{\text{inel}}^{\text{NN}} = 6.18\,\text{fm}^2$	ζ/s: Lorentzian shape	parameters: T_{chem}, T_{kin}
scaling: $Norm/\tau_0$	parameters: η/s, $(\zeta/s)_{\text{max}}$	
parameters: $Norm$, τ_0		

TABLE 5.1: Summary of the settings and assumptions for modeling the heavy-ion collision.

For the particle decays, only strong and electromagnetic decays of $\sim$ 700 resonances with masses up to $m \approx 3\,\text{GeV}$ are considered, the settings are taken from ref. [24].
In contrast to the previous study in [24], this analysis differentiates between the chemical and the kinetic freeze-out by adding a phase of partial chemical equilibrium to the model. The freeze-out temperatures T_{chem} and T_{kin} are taken as free parameters of the model that will be optimized.
A summary of the parameters used by the model of the particalization and resonance decays is given in table 5.1.

Comparing to experimental data

In this work, the comparison between the fluid dynamic simulation and the experiment will be done based on the transverse momentum spectra of pions, kaons, and protons in five centrality classes of Pb–Pb collisions at the center-of-mass energy per nucleon pair $\sqrt{s_{NN}}$ =2.76 TeV. The considered momentum range is $p_{\text{T}} < 3\,\text{GeV/c}$. The data differs from previous analyses of model-to-data fits, where momentum-integrated quantities, like particle multiplicity, mean transverse momentum, or flow harmonics were used [23]. Nevertheless, as argued in ref. [24], the particle spectra provide sufficient information to also set constraints on the transport properties, especially because the experimental data available is of high precision.
The experimental data that will be used for the analysis was measured by the ALICE collaboration in the 2010 run at the LHC [52]. The data comprises the p_{T}-spectra of pions (π), kaons (K) and protons (p) for the centrality classes 0-5%, 5-10%, 10-20%, 20-30%, and 30-40% in the p_{T}-range $<$3 GeV/c, measured at mid-rapidity ($|\,y\,| <$0.5). They are given in the form $\frac{1}{N}\frac{1}{2\pi p_{\text{T}}}\frac{dN}{dp_{\text{T}}dy}$.
The experimental data is shown in fig. 5.2. For pions, the considered p_{T}-range will be modified to 0.5-3 GeV/c. This is done because previous analyses [24] showed that the current fluid dynamic modeling significantly underestimated yields of low p_{T}-pions, suggesting physical processes that are not included in the model. Therefore they will also not be included in the fit.

5.2 Model properties

After the simulation has been defined, particle spectra can be obtained from it by fixing the free parameters and running its modules subsequently. An example of the output of the model for an arbitrary set of input parameters is shown in fig. 5.3. The similarity to the experimental data in fig. 5.2 is apparent although the parameters are not even optimized yet.

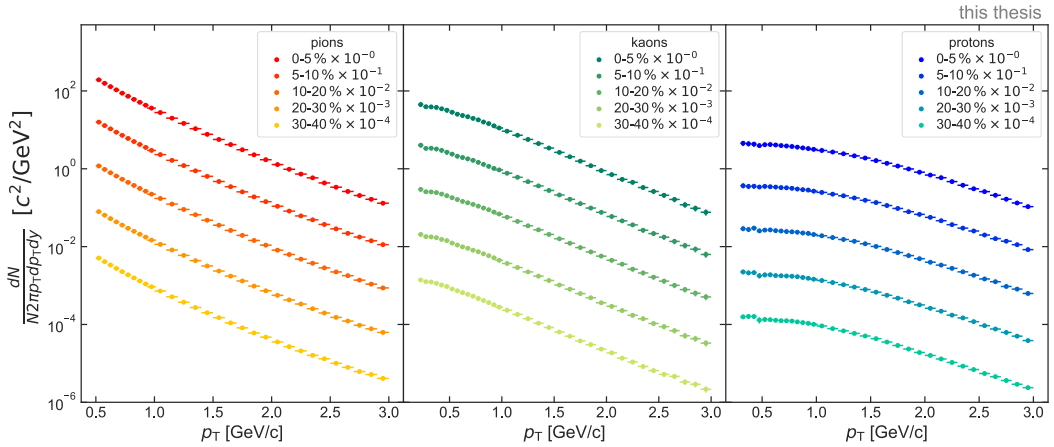


FIGURE 5.2: p_T-spectra for pions (left), kaons (middle) and protons (right) measured by ALICE in Pb-Pb collisions at $\sqrt{s_{NN}}$ =2.76 TeV.

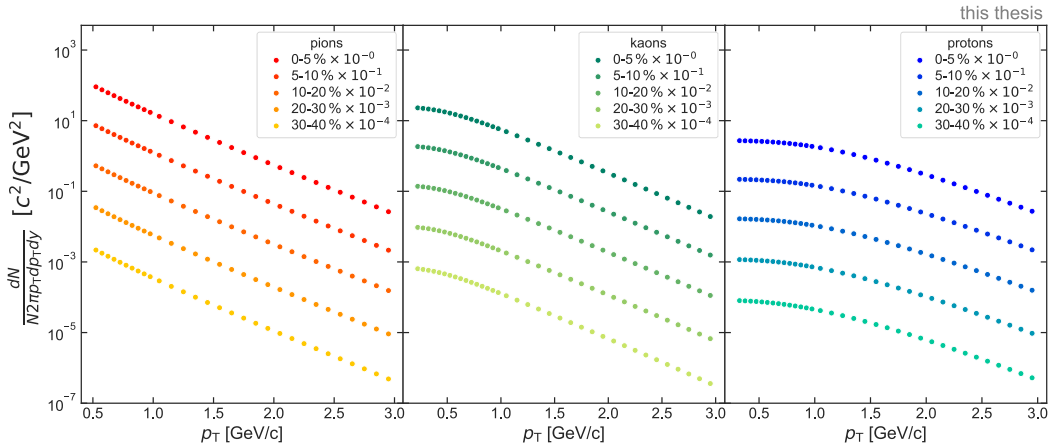


FIGURE 5.3: p_T-spectra for pions, kaons and protons produced by a combination of Trento, Fluidum and FastReso for an arbitrary set of parameters $Norm = 18.3$, $\eta/s = 0.69$, $\zeta/s = 0.003$, $\tau_0 = 0.57\,\mathrm{fm/c}$, $T_{\mathrm{kin}} = 128.2\,\mathrm{MeV}$ and $T_{\mathrm{chem}} = 145.5\,\mathrm{MeV}$.

Setting the free parameters and executing the model can be conceptualized as evaluating a function. In this description, the function f would be represented by the model combination of Trento, Fluidum and FastReso, which takes a set of parameters $Norm$, η/s, ζ/s, τ_0, T_{kin} and T_{chem} as an input $\mathbf{x}$ and returns the p_T-spectra of pions, kaons and protons as an output $\mathbf{y}$. The concept is depicted in fig. 5.4. Since this description of the problem is quite general and beneficial for the analysis, the parameters and the spectra will be often referred to by $\mathbf{x}$ and $\mathbf{y}$ in the following.
To decide on a proper procedure for the analysis, it is important to discuss the properties of the model f and its parameters first. Three main features can be identified, which will be discussed in detail in the following:

1. The simulation is a computationally expensive black-box function.
2. The model can be considered deterministic.
3. The response of the model is continuous with respect to its input parameters.

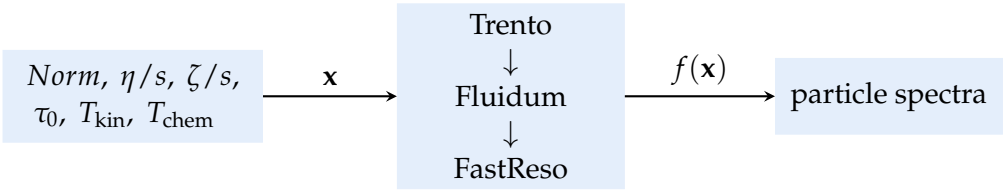
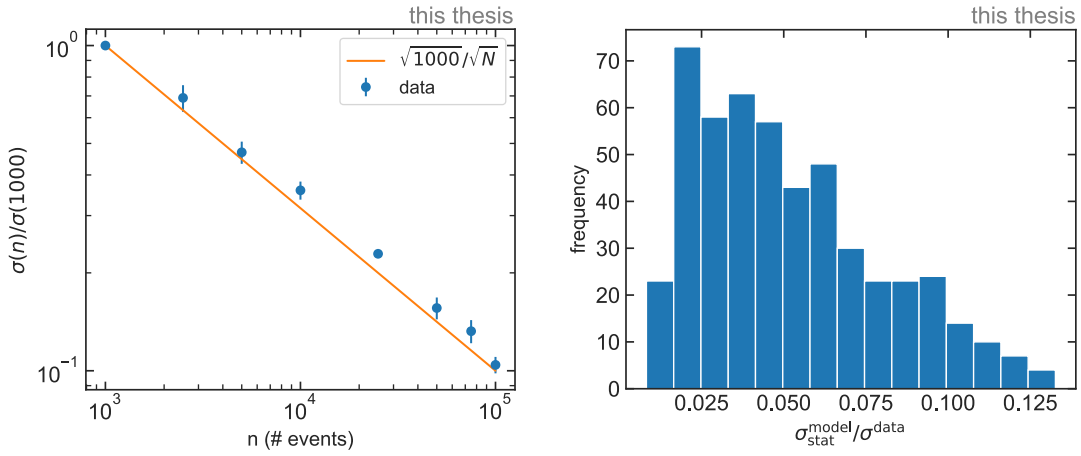


FIGURE 5.4: The model evaluation can be conceptualized by a function call $\mathbf{y} = f(\mathbf{x})$, taking input parameters $\mathbf{x}$ and returning particle spectra $\mathbf{y}$.

The first point is true because although each step of the model is in principle traceable by physical or phenomenological relations, the global behavior and the interactions between different parameters are not. How a change in a parameter affects the output underlies complex inter-dependencies between the different parts thus making it impossible to predict how the model reacts to different inputs. Moreover, no analytical expression for the model exists, such that the impact of individual parameters is not accessible via mathematical calculations. Therefore it can be seen as a black-box function. Additionally to this, the model is computationally rather expensive because of the numerical solving scheme for the partial differential equations of the fluid dynamic evolution. Although Fluidum is much faster than event-by-event simulations, it still takes about 3 min of computation time to produce particle spectra output for one set of parameters. This may seem fast, however, a large number of model evaluations is needed to infer the optimal set of parameters. This problem may be circumvented by massive parallelization of the model evaluations, but due to computational limitations[1] this is not possible here. This makes it even more important to build an efficient procedure to estimate the physical properties of the model.
The second feature considers the question of whether the model is statistical in its outputs for a given input or if it is fully deterministic. From its construction, it is obvious that the initial state model Trento is statistical since it is an MC model, therefore also outputs of the full model will fluctuate, even though Fluidum and FastReso are deterministic. However, in the calculation of the spectra, only the background field is considered, which is built by averaging a large number of events. By this, the statistical variations of the initial state are averaged out to a large degree, in the limit of infinitely many events the fluctuations would converge to zero making the full model completely deterministic again. To investigate how large the fluctuations in the spectra are with respect to the initial state, the full model is evaluated 100 times for the same random set of input parameters for different numbers of events ranging from 10^3 to 10^5. For each number of events, the fluctuations are quantified by the standard deviation of the spectra values from the 100 runs. This gives a measure for the fluctuations for each p_T-bin for each particle and for each centrality class for the respective number of events n. To visualize the decrease of the fluctuations for an increasing number of events, the measure of the fluctuations is normalized with their respective value at $n = 1000$. Thus, for each p_T a value of the relative fluctuation is derived. Since considering the relative decrease eliminates any dependencies on momentum, centrality, and particle species, all measures of the fluctuation for a specific number of events n can be averaged and an uncertainty can be estimated by the standard deviation. The result is shown in fig. 5.5A.
It is evident, that the fluctuations decrease with $1/\sqrt{n}$, as is expected for the statistical uncertainty. For an increase in the number of events from 10^3 to 10^5, the statistical error

[1] Fluidum is written in Mathematica and only 21 licenses for it are available on the used computing cluster.



(A) Average decrease in the relative uncertainty of the spectra values. All p_T-bins for the spectra of one n are averaged. The $1/\sqrt{n}$ dependence is clearly visible.

(B) Distribution of the ratio between the statistical uncertainties of the model for 10^5 events and the data uncertainty. The histogram combines the ratios of all p_T-bins for all particles.

FIGURE 5.5: Statistical fluctuations.

hence decreases by a factor of 10. Although the behavior of the fluctuations is only determined for one set of parameters, it can be assumed, that it is general, since the parameters do not affect the initial state.
What still has to be answered is the question if the absolute value of the statistical uncertainty is sufficiently small to be negligible in the analysis if $n = 10^5$ as chosen in 5.1. For that, the statistical uncertainty can be compared to the experimental uncertainty. In fig. 5.5B, the distribution of the ratio between the statistical uncertainty from the model and the data uncertainty is shown. In the histogram, the values of all p_T-bins for all particles and in all centrality classes are displayed together. The statistical uncertainties are always below 13 % of the data uncertainty, most of them are around 4%. This is sufficiently small to impact the analysis only marginally. Therefore, the model is approximately assumed to be deterministic.
The last feature that is considered here is how the model reacts to a variation of its parameters. The input parameters themselves are continuous, but the output may still exhibit discontinuities which would have a large impact on the procedure for the analysis. However, from the construction of the model, it can be assumed, that it produces continuous outputs with respect to the inputs because the initial state is not impacted by a change of parameters, the fluid evolution is differentiable and therefore continuous and the particalization model calculates averaged p_T-spectra and is therefore continuous, too. Nevertheless, the continuity was checked for a random set of input parameters by varying one parameter at a time and observing the effect on the particle spectra. The results are depicted in fig. 5.6. What is visible is the change of the first p_T-bin spectra value with respect to a change in one input parameter while keeping the other parameters fixed. It is clearly visible that the response to a change in the parameters is continuous. This was also checked for all other p_T-bins.

5.3 Finding the optimal parameters

After having characterized the model and the experimental data, the time has come to address the main goal of this work, the determination of the optimal parameters for the model to reproduce experimental data. To reach this goal, it has to be at first defined

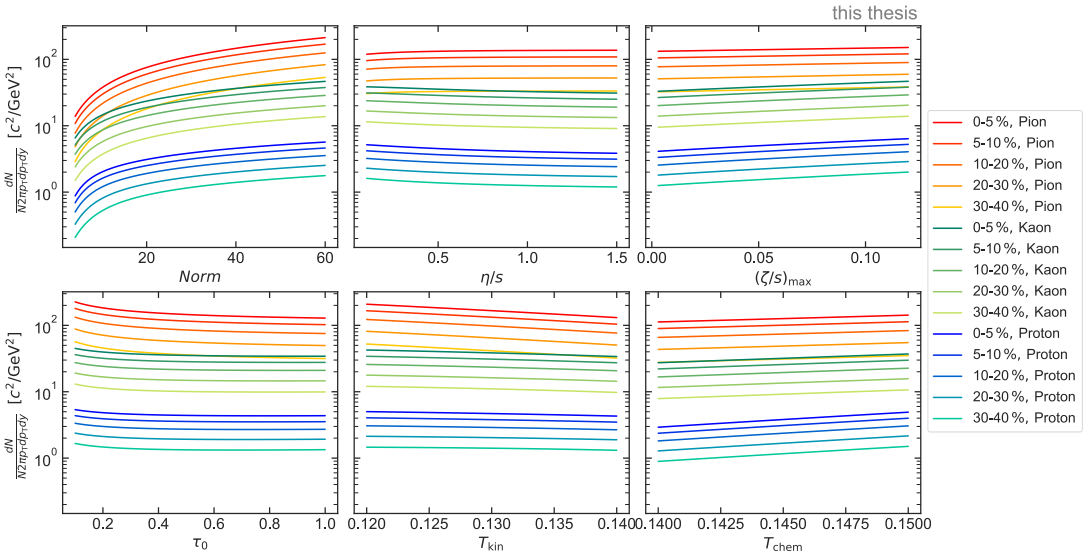


FIGURE 5.6: Spectra values of the first p_T-bin of all particles and centrality classes in dependence of a variation in one parameter at a time. The dependence is continuous.

how *optimality* of the parameters can be quantified. Depending on the measure of the optimality, the resulting set of parameters could vary largely. A reasonable choice that was used in [24], the predecessor of this work, is the χ^2-value between model output and data, which is given by [24]

$$\chi^2 = \sum_{i=1}^{N} \frac{(x_i - y_i)^2}{\sigma_i^2}, \tag{5.5}$$

where x_i is the experimental value of the transverse momentum spectrum of a particle species in a specific p_T-bin and centrality class, y_i is the corresponding model output for a fixed set of parameters, and $\sigma_i = \sqrt{\sigma_{i,\text{sys}}^2 + \sigma_{i,\text{stat}}^2}$ is the square-root of the sum of squares of the systematic and the statistical uncertainties of the experimental data. The sum is taken over all considered p_T-bins in all centrality classes for all particles. To find the optimal parameters Devetak et al. minimized the χ^2-value. In practice, this was realized by generating a 6D parameter grid and then calculating the χ^2 for each set of parameters in the grid. The resulting χ^2-grid was interpolated by a spline model and minimized by a minimization algorithm. This approach is straightforward, however, it has some disadvantages:

- In the experimental data, the systematic uncertainties are usually correlated. In the χ^2-value, this is not considered.
- Spline interpolation is not accurate if the distances between the data points are large. But in high-dimensional spaces, the density of points will be low such that the interpolation uncertainties increase. These uncertainties were not considered in the analysis of [24].
- Correlations between parameters were calculated by 2D slices and fitting a quadratic function to the minimum, however, it is not clear a priori, that the functional dependence is quadratic.

- For uncertainty quantification no correlations between the parameters were considered, they were obtained by 1D slices of the χ^2-landscape, making them potentially smaller than they are.

To overcome these limitations and to improve the analysis performed in [24], in this work, the optimality criterion will be based on Bayesian inference. Bayesian inference offers a natural framework to optimize model parameters and quantify their uncertainties and was used already extensively for such tasks, for example in [5] in the context of heavy-ion collisions.

5.3.1 Bayesian inference

In Bayesian inference, posterior probability distributions for each model parameter are inferred, which quantify how likely each parameter value is given the experimental data and the model. To understand how to compute these posterior densities, assume a set of n parameters $\mathbf{x} = (x_1, x_2, ..., x_n)$ and observed experimental and model data $\mathbf{D}$. Then, the so-called *prior* probability distribution $p(\mathbf{x})$ can be defined, which represents the initial beliefs about the likeliness of each parameter value. Moreover, the likelihood $p(\mathbf{D}|\mathbf{x})$ is introduced, which quantifies how likely it is to observe the data $\mathbf{D}$ given a specific set of the parameters $\mathbf{x}$. This is connected to the quality of the fit of the model to data. From these probabilities, the posterior probability distribution $p(\mathbf{x}|\mathbf{D})$ can be calculated by Bayes formula:

$$p(\mathbf{x}|\mathbf{D}) = \frac{p(\mathbf{D}|\mathbf{x})p(\mathbf{x})}{p(\mathbf{D})}, \tag{5.6}$$

where $p(\mathbf{D}) = \int p(\mathbf{D}|\mathbf{x})p(\mathbf{x})d\mathbf{x}$ is the marginal likelihood describing how likely an observation of $\mathbf{D}$ is under the prior beliefs. This term can be neglected, since it does not dependent on the parameters $\mathbf{x}$ and therefore only acts as a normalization constant. The resulting posterior probability distribution $p(\mathbf{x}|\mathbf{D})$ is n-dimensional, however, to infer estimates of the uncertainties of the parameters, usually, the marginal probability densities are used. They are constructed by marginalizing over (integrating out) all but one parameter x_i of $p(\mathbf{x}|\mathbf{D})$. The marginal posterior probability distribution of x_1 would be for example given by

$$p(x_1|\mathbf{D}) = \int dx_2 ... dx_n p(\mathbf{x}|\mathbf{D}). \tag{5.7}$$

This distribution quantifies the probability for each parameter value. From that, uncertainties may be estimated by considering measures of the width of the distribution like credible intervals.
For the problem at hand, the parameters $\mathbf{x}$ can be identified with six input parameters of the simulation, and the data $\mathbf{D}$ is given by the experimental spectra as well as the model output spectra. In the following, the form of the prior distribution $p(\mathbf{x})$ and likelihood function $p(\mathbf{D}|\mathbf{x})$ are discussed for this case.

Choice of prior

The prior $p(\mathbf{x})$ contains any information about the parameters of the model before observing any data. In the present case, little information about the parameters is available. Therefore, a constant prior is an appropriate choice, such that every parameter combination is assumed to have a priori an equal probability to be the optimal set of parameters. Because the search range of the parameters will be finite in the inference, it is reasonable to restrict also the prior to these ranges. With that, the prior is given as a uniform distribution:

$$p(\mathbf{x}) \propto \begin{cases} 1 & \text{if } \min(x_i) \leq x_i \leq \max(x_i) \text{ for all } i \\ 0 & \text{else} \end{cases}, \tag{5.8}$$

where x_i are the input parameters. Setting the prior outside of the search ranges to zero is a strong assumption. By that, it is assumed that there is no probability to observe any such parameter value. To avoid excluding plausible values outside of the search region it is therefore important to define the parameter ranges large enough to include all reasonable values.
The choice of the prior is to some extent arbitrary here since only little information about the parameters is given beforehand. It could be also assumed to be of another form, the only requirement that needs to be fulfilled is that it is *uninformative*. This does not mean that the prior contains no information, but that it impacts the inference only marginally. The posterior distribution obtained from eq. 5.6 is determined for an uninformative prior mainly by the likelihood $p(\mathbf{D}|\mathbf{x})$ if this is well constrained by the data. In this case, the Bayesian approach yields not too different results from conventional statistical analysis.

Likelihood function

The likelihood $p(\mathbf{D}|\mathbf{x})$ is the probability of observing the data $\mathbf{D}$ given the parameters $\mathbf{x}$. $\mathbf{D}$ includes the experimental data as well as the data generated by the model with the set of parameters $\mathbf{x}$. In a sense, $p(\mathbf{D}|\mathbf{x})$ can be seen as the fit probability of the model outcome to the experimental data, thus quantifying the goodness of the fit.
To infer the form of the likelihood function, it is convenient to define a few terms at first. The following calculations are mostly adapted from [5]. Let the experimental data be given by the vector $\mathbf{y}_\mathrm{e}$, which contains all spectra values of pions, kaons, and protons of all considered centrality classes in all p_T-bins in its elements. Then $\mathbf{y}_\mathrm{e}$ may be represented by the sum of the physically "true" data $\mathbf{y}_\mathrm{e}^\mathrm{true}$ and some experimental uncertainty $\boldsymbol{\epsilon}_\mathrm{e}$, such that

$$\mathbf{y}_e = \mathbf{y}_e^\mathrm{true} + \boldsymbol{\epsilon}_e, \quad \boldsymbol{\epsilon}_e \sim \mathcal{N}(\mathbf{0}, \Sigma_e). \tag{5.9}$$

The experimental uncertainty $\boldsymbol{\epsilon}_\mathrm{e}$ is assumed to be a multivariate Gaussian with zero mean and a covariance matrix Σ_e, which accounts for statistical and systematic uncertainties. The simulation output can be represented in the same form, leading to

$$\mathbf{y}_m(\mathbf{x}) = \mathbf{y}_m^\mathrm{ideal}(\mathbf{x}) + \boldsymbol{\epsilon}_m, \quad \boldsymbol{\epsilon}_m \sim \mathcal{N}(\mathbf{0}, \Sigma_m), \tag{5.10}$$

where $\mathbf{y}_m^\mathrm{ideal}(\mathbf{x})$ describes the outcome of a hypothetical ideal model, which does not have any statistical or systematic uncertainties, and $\boldsymbol{\epsilon}_m$ is added to account for these uncertainties. $\boldsymbol{\epsilon}_m$ is assumed to come from a multivariate normal distribution with covariance matrix Σ_m. Assuming there is a set of optimal parameters $\mathbf{x}_\star$, such that $\mathbf{y}_e^\mathrm{true} = \mathbf{y}_m^\mathrm{ideal}(\mathbf{x}_\star)$, combining eqs. 5.9 and 5.10 gives

$$\mathbf{y}_m(\mathbf{x}_\star) - \mathbf{y}_e = \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \Sigma), \quad \Sigma = \Sigma_e + \Sigma_m. \tag{5.11}$$

From this the likelihood can be constructed, which is given by a Gaussian distribution with a mean $\mathbf{y}_m(\mathbf{x}) - \mathbf{y}_e$ and a covariance Σ

$$P(D \mid \mathbf{x}) = \frac{1}{\sqrt{(2\pi)^n \det \Sigma}} \exp\left\{-\frac{1}{2}\left[\mathbf{y}_m(\mathbf{x}) - \mathbf{y}_e\right]^\mathrm{T} \Sigma^{-1} \left[\mathbf{y}_m(\mathbf{x}) - \mathbf{y}_e\right]\right\}, \tag{5.12}$$

where n is the number of parameters. By eq. 5.12, the likelihood can be computed for any fixed set of input parameters $\mathbf{x}$. For that, the simulation has to be evaluated and the covariance matrix has to be constructed. Eq. 5.12 can be further simplified if the model uncertainty as well as correlations in the uncertainties are neglected ($\Sigma = \Sigma_e$, Σ diagonal). Then the term in the exponential reduces to the χ^2-value from eq. 5.5, such that minimizing the χ^2-value coincides with maximizing the likelihood

$$P(D \mid \mathbf{x}) \propto \exp\left\{-\frac{1}{2}\chi^2\right\}. \tag{5.13}$$

5.3.2 General procedure

With the prior probability and the likelihood function in place, following eq. 5.6, the posterior probability can be computed for any fixed set of parameters $\mathbf{x}$. However, to infer the full probability distribution, the posterior probability needs to be known for *any* set of parameters in the parameter space. An analytic treatment would be the best solution to this, but since the likelihood is a black-box function due to the model evaluation in it, it is not applicable here. Thus numerical methods have to be applied to approximate the posterior distribution.
The method that will be used here to populate the probability space of the posterior is the MCMC algorithm that was already introduced in 4.3. It provides an efficient way to obtain the probability distribution also in the high-dimensional parameter space of the problem. Within the algorithm, random walks through the parameter space are generated which are governed by the posterior probability. This is done by sampling a random chain of steps and calculating the posterior probability as the target probability for the Metropolis criterion (eq. 4.10) for each step. By construction, the distribution of the parameter sets in the chains will then converge to the true posterior distribution.
However, this procedure involves a computational difficulty. Since the chains are constructed by sequentially computing the posterior probability for thousands of sets of parameters, also the model has to be evaluated thousands of times while taking approximately 3 min per evaluation. In a reasonable amount of time, it is not possible to obtain meaningful posterior estimates with this procedure because of the slow evaluation. To overcome this obstacle, an emulator model can be introduced. The purpose of this model is to emulate the computationally expensive simulation, such that it returns the same outputs for the same inputs. The idea is that it does this much faster, taking only a fraction of a second for a model evaluation, lowering the run-time of the MCMC method to a reasonable value.
To construct an emulator model, input-output pairs can be generated with the simulation, which are then used as training samples for the emulator model. Technically, this is a supervised machine learning problem as introduced in chapter 4, more precisely it is a regression problem since the output variables are continuous. The emulator is therefore a regression model, which interpolates between data points obtained from the true simulation. To train the emulator, input-output samples are usually generated by evaluating the model on a grid structure in the parameter space. This step is again computationally expensive since the model has to be evaluated many times, however, it can be parallelized since the points in the parameter grid are independent of each other.
Considering all of the above-mentioned steps, the workflow for inferring the posterior probability distribution can be summed up as follows:

- **Parameter grid** Define parameter ranges and generate a grid of parameter sets $\mathbf{x}_i$ within the parameter ranges. Run the simulation (Trento, Fluidum, FastReso) for every grid point to infer the model outputs $\mathbf{y}_i$.
- **Emulator model** Define an efficient emulator model and use the input-output pairs $\{\mathbf{x}_i, \mathbf{y}_i\}$ to train the emulator to reproduce the model outputs. Verify the performance of the emulator and quantify the uncertainty of the emulator.
- **Bayesian inference** Run the Bayesian inference to infer the posterior probability distributions for each parameter. This is done by employing the MCMC algorithm. Instead of evaluating the model, the emulator is used. The posterior is calculated according to eq. 5.6.
- **Parameter estimates** Obtain parameter estimates by computing the marginal posterior probability for each parameter. Parameter correlations can be inferred from joint distributions.

The procedure is also illustrated in fig. 5.7. In the next sections, each of these steps will be executed for the considered design of the study and the approach to each of them will be discussed in detail.

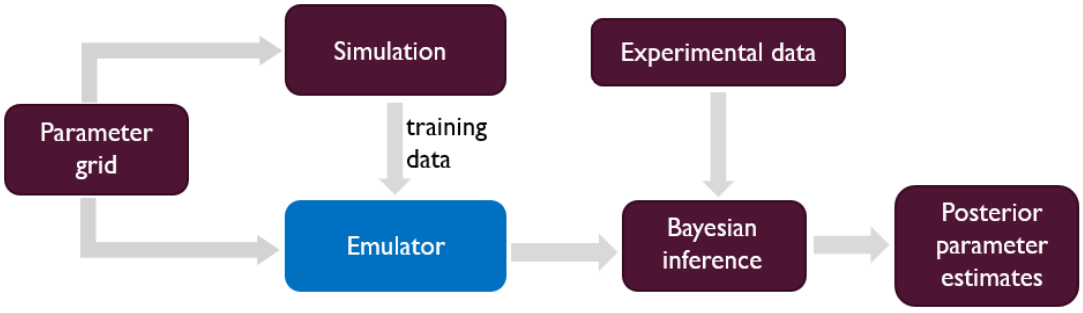


FIGURE 5.7: General procedure for inferring the posterior parameter estimates in a Bayesian analysis.

5.4 Parameter grid

In the first step of the procedure, a parameter grid has to be defined which can then be used to obtain training data for the emulator model. Its design is of particular importance since it has a large impact on the performance of the emulator. If the training data does not contain sufficient information, the uncertainties of the emulator will increase and thus the uncertainties of the posterior. This is especially an issue in high dimensions where the density of points in the parameter space decreases exponentially with the dimension for a fixed number of grid points. Because the number of possible model evaluations is limited by the computational cost, the design of the grid has to be chosen such that the maximal information is inferred from a minimal number of grid points.
For the choice of the grid design, there are three different possibilities. All of them generate grid points in a hypercube inside the parameter space that is defined by the search ranges for the respective parameters. The most simplistic approach would be a factorial design with k values for each of n dimensions. The advantage is that the local density is the same everywhere and thus the emulator will have a similar uncertainty everywhere. However, the disadvantage is that the number of grid points increases exponentially with k^n, which quickly reaches the computational limits of the model

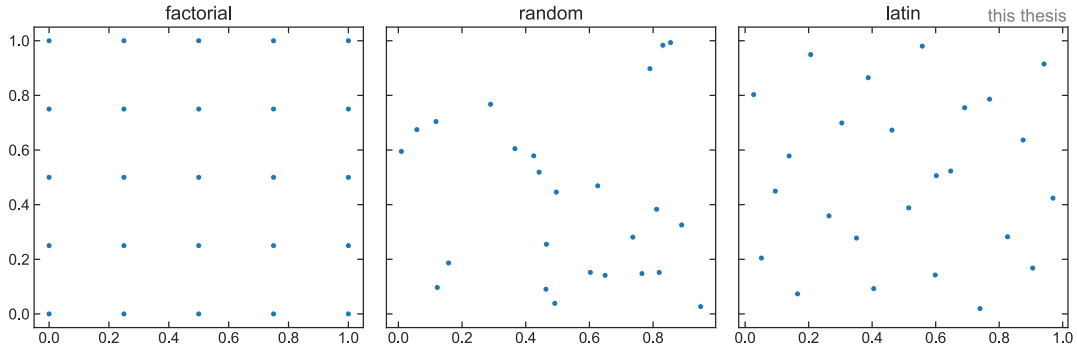


FIGURE 5.8: Depiction of a factorial, a random, and a Latin grid design. The Latin grid combines the homogeneity of the factorial design and the randomness of the random grid.

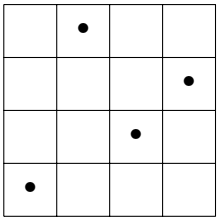


FIGURE 5.9: Example of Latin hypercube sampling. In every row and every column, there is exactly one data point.

evaluations even for small k. In addition, the grid contains only little information since for every parameter only k values are tested.
This can be circumvented by using a random grid. For this, N grid points are sampled uniformly and randomly in the high-dimensional search space, thus all their parameter values will be most likely different from each other. Disadvantageously to this approach is, that the local density of points may vary due to random sampling, resulting in well and poorly-populated areas.
The method that can resolve both problems is Latin hypercube sampling. Within this method, points are sampled semi-randomly in a n-dimensional unit hypercube $[0,1]^n$ that can then be scaled to the desired search region. The idea of this method is to divide each parameter axis into N equal probable intervals and then place N sample points in the grid in such a way, that exactly one point is in each interval. The approach is illustrated for the 2D case in fig. 5.9. The axes are divided into four rows and four columns and in each row and each column exactly one point is sampled. To ensure that the density of points is the same everywhere, the method can also be combined with additional optimization methods.
Before a parameter grid can be generated, the parameter search ranges must be defined. They are chosen in this work according to tab. 5.2.

$Norm$	η/s	$(\zeta/s)_{\max}$	τ_0 [fm/c]	T_{kin} [MeV]	T_{chem} [MeV]
$1-60$	$0.1-1.5$	$0.003-0.12$	$0.1-1.0$	$120-140$	$140-150$

TABLE 5.2: Search region for the parameters.

These search regions are based on physical considerations as well as previous analyses [23], [24]. The $Norm$ parameter has the least constraints since it is not related directly to any physical interpretation. The shear and bulk viscosities are expected to be rather

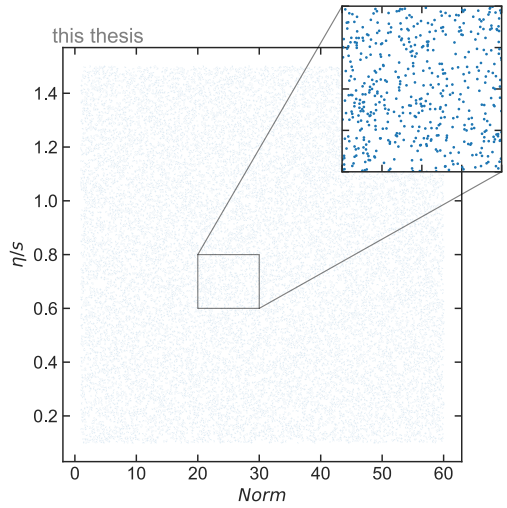


FIGURE 5.10: Two-dimensional projection of the used parameter grid. The homogeneity and randomness of the Latin approach are apparent.

small because previous analyses showed that the QGP is an almost perfect fluid [21]. For τ_0, T_{kin} and T_{chem}, the search ranges can be chosen based on physical observables. T_{kin} and T_{chem} are expected to be in the order of magnitude of the critical temperature T_{c} and τ_0 is expected to be below 1 fm/c.
Having defined the parameter ranges, a Latin hypercube grid is constructed. The number of points is chosen to be $N = 20000$, which is a compromise between density in the parameter space and computation time. Potentially, numbers up to $N = 10^5$ would be feasible in a reasonable amount of time. Additionally, a second Latin hypercube grid is sampled within the same ranges with $N = 4000$. This grid is used partly for testing and validating the emulator model. In fig. 5.10, a two-dimensional projection of the generated training grid is visible. It is apparent, that the density of points is homogeneous in the whole search region. The grid looks densely packed, however, this is due to the projection onto two axis, in the six-dimensional space neighboring points may be far apart. After the grid is built, the simulation with Trento, Fluidum, and FastReso is run for all grid points to obtain the model spectra. In this case, the Trento output can be pre-computed and used for all grid points because the initial state parameters are fixed. For the given parameter grid, the spectra calculations succeeded for 19542 of the 20000 design points. The failed computations are connected to specific parameter combinations that lead to numerical issues in Fluidum. Since the number of failed calculations is rather small, these parameter combinations are omitted here.

5.4.1 Grid validation

To get a first impression of whether the chosen parameter ranges include the optimal set of parameters to reproduce the experimental data, the computed spectra can be plotted together with the data in one figure. This is shown in fig. 5.11.
The variety of the spectra for the different parameter configurations of the grid is depicted by box plots. For each p_{T}-bin, the boxes indicate the lower (Q1) and the upper quartile (Q3) values of the observed spectra values for the grid, with an additional line at the median. The range between the two quartiles (Q3-Q1) is also called the interquartile range (ICR). The whiskers of the boxes start from the first datum larger than $\mathrm{Q1} - 1.5\,\mathrm{ICR}$ to the last sample less than $\mathrm{Q3} + 1.5\,\mathrm{ICR}$, following the standard definition of box plots. Outliers are not shown to improve readability.

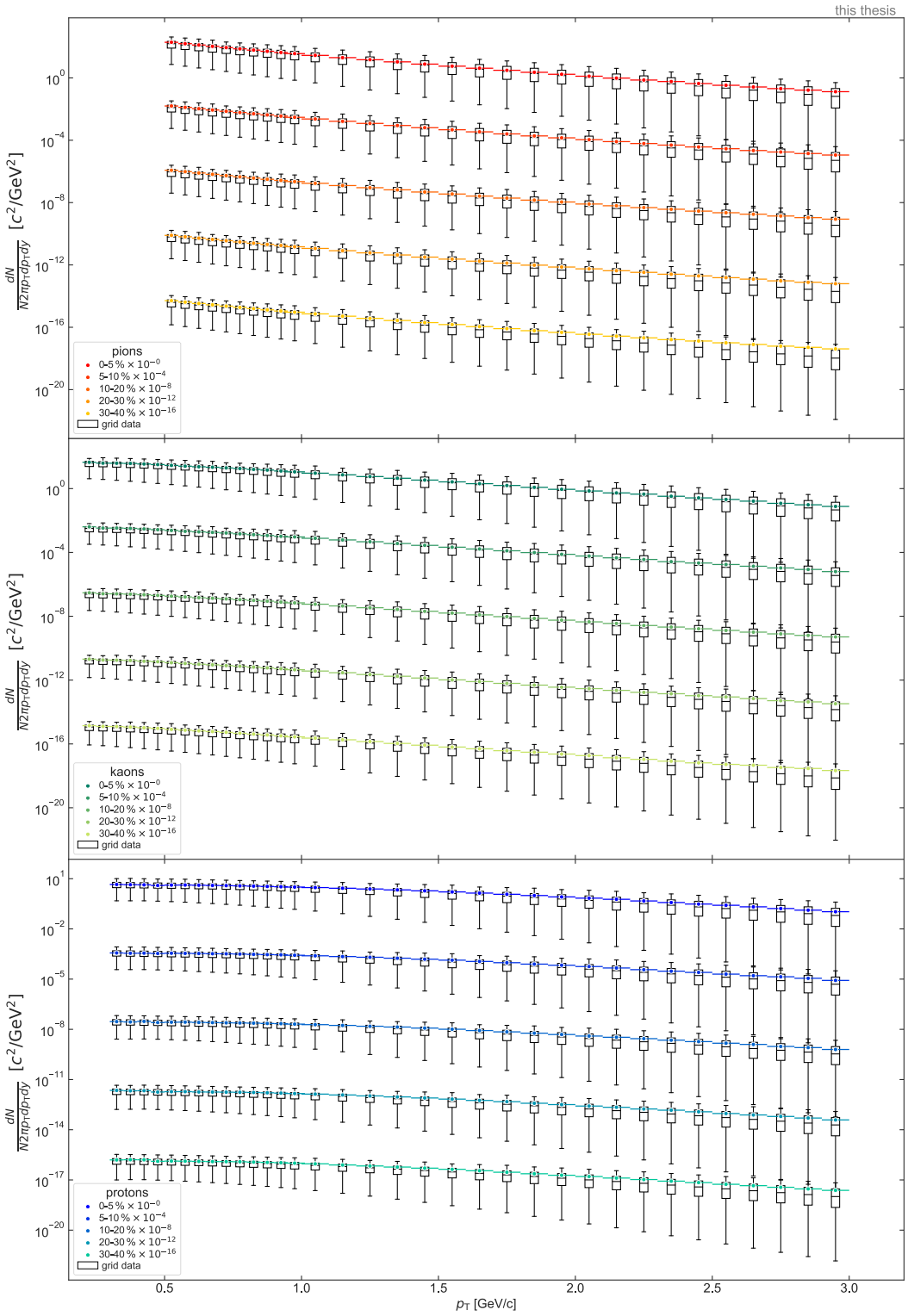


FIGURE 5.11: Distribution of the spectra values generated with the parameter grid compared to the experimental data. The boxes extend from the lower to the upper quartile values of the data, with a line at the median. It is obvious, that the experimental data is captured within the grid data. Note that the y-axis is in log scale and that the scaling factors deviate from previous plots to improve the visualization.

From the figure, it is visible that the spectra generated with the parameter grid are distributed around the experimental spectra. For all p_{T}-bins except for high-centrality, high-p_{T} pions, the experimental data is captured within the ICR of the grid data. This indicates that good parameter estimates may be found within the search region to reproduce the experimental data.

5.5 Emulator model

The next step in the proposed procedure in sec. 5.3.2 is the construction of an emulator model, which utilizes the generated input-output pairs $\{\mathbf{x_i}, \mathbf{y_i}\}$ from the parameter grid to reproduce the simulation. Because this problem is a standard machine learning regression task, there are many available options for the emulator model. The canonical choice is Gaussian process regression [53], which is widely used for interpolation and was also employed for parameter estimation in the field of heavy-ion physics [23]. It is a statistical non-parametric method that can interpolate multi-dimensional functions, exactly what is needed here. The method also provides naturally a Gaussian uncertainty estimate for its prediction. However, there are a few caveats for the emulation in this case: Besides being optimal for computationally very expensive models where only a few hundred data points can be generated, for less demanding models Gaussian process regression is inefficient because the method scales with $\mathcal{O}(n^2)$ in memory and $\mathcal{O}(n^3)$ in computation. In this analysis, the number of training samples n is $\sim 10^4 - 10^5$, leading to already enhanced computation times. Furthermore, Gaussian process regression can only return scalar outputs, thus for multi-dimensional outputs like in the present case, additional methods have to be built on top to convert the scalar outputs to multi-dimensional ones.
Because of these reasons, in this thesis, a different method will be used for emulation, namely neural networks. NNs are, if constructed large enough, as flexible as Gaussian process regression and are thus able to fit multi-dimensional functions similarly good. In fact, it has been shown that infinitely wide NNs can converge to Gaussian process regression [54]. The advantage of NNs is the good scaling with the number of training samples. The computation time for the training scales with $\mathcal{O}(n)$, whereas the model's memory consumption is not affected by the number of samples. Furthermore, NNs can be constructed to process any number of inputs and outputs by adjusting the number of input and output nodes. With that, the multi-dimensional output can be fitted directly without further treatment. The only disadvantage that is connected to the use of NNs is, that they do not quantify their uncertainty but only give point estimates. To obtain an uncertainty estimate, additional methods must be laid out. In this work, an ensemble of NNs will be used for that. The multiple NNs in the ensemble vary in their predictions, hence their spread can be used as an estimate of the model uncertainty. In the following, at first, the data preprocessing is discussed before a NN model is constructed and verified and the ensemble is introduced and verified as well.

5.5.1 Data preprocessing

Before the NN emulator model is constructed, the data obtained from the parameter grid is preprocessed. In the first step, the data is split into a training, a validation, and a test set. This is done to have independent testing data for validating the model. The training set is built with the 19542 input-output pairs from the first parameter grid and the validation and test sets are built by splitting the 3909 data pairs of the second parameter grid in half. With that, the split of training, validation, and test set is 83.3 % to

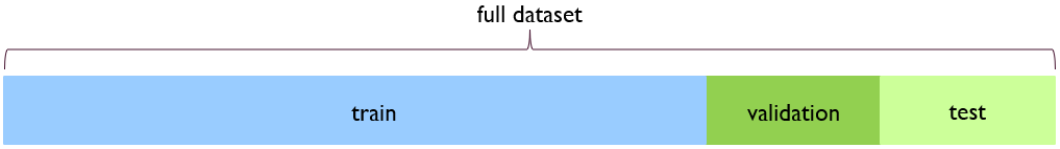


FIGURE 5.12: Illustration for the splitting of the data into a train, validation, and test set.

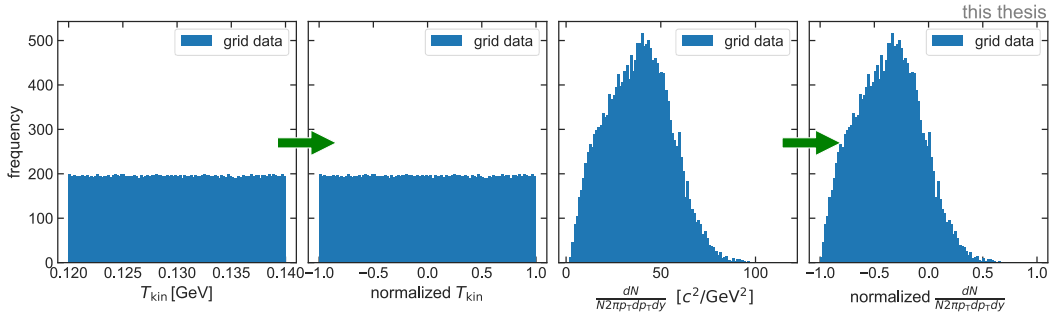


FIGURE 5.13: Examples of the normalization procedure. Left: Input parameter is normalized to the range [-1, 1]. Right: An output variable is normalized to the same range.

8.3 % to 8.3 %, similar to fig. 5.12. The training set is used only for training the emulator model, whereas the validation set is used for testing the model while optimizing its parameters, and the test set is used for the final performance quantification. The distinction between validation and test set is made because the validation error underestimates the true error of the model.
Additionally, the data is normalized based on the training data. For that, every input parameter is scaled to the range [-1,1], such that the minimum value of the parameter is converted to -1 and the maximum to 1. Because this normalization is done based on the training set, -1 and 1 will not correspond exactly to the grid's boundaries, however, will be very close to it. For example, the corresponding values for the *Norm*-parameter are 1.00388 and 59.9994.
There are two reasons for normalizing the input parameters: The first is the different magnitude of the parameters. While the range for *Norm* is $1 - 60$, it is $0.14 - 0.15$ for T_{kin}. This impacts the gradient descent algorithm, such that parameters with a larger magnitude will be given more importance in the learning process, which may worsen the result. The second reason is, that the activation functions operate in the region around zero, thus if the parameter values are far away from it, the gradient will be very small and training is inefficient. This is related to the problem of vanishing gradients, where gradients calculated by backpropagation get smaller while propagating through the network until they vanish completely. At that point, the network does not train at all. Normalization may prevent this problem.
The outputs are normalized to the range [-1, 1] for the same reason. This is done for the spectra values of each p_{T}-bin in each centrality class for each particle. Examples of the normalization procedure are shown in fig. 5.13.
The normalization of in- and outputs leads to a significant increase in the convergence rate of the training process and makes the training much faster. The procedure will be included in the emulator model, such that given parameter inputs will be normalized automatically, then fed through the networks, and then the resulting outputs will be scaled back to the output ranges.

5.5.2 Neural network construction and training

Having normalized the in- and outputs, the NN emulator model may be constructed. For this, all hyperparameters that were introduced in sec. 4.1.2 have to be specified, which is elaborated on in the following:

- **Number of layers and number of nodes per layer** The NN has to take six input parameters and return the spectra values of pions, kaons, and protons in five centrality classes, which add up to 500 output values. Thus the network has to have six nodes in the input layer and 500 in the output layer. The size and number of hidden layers are not as well defined, the optimal value for them will be inferred in a grid search. For this, a NN with 1024 hidden nodes is considered, as first tests indicated that this is enough to reproduce the complexity of the simulation. These 1024 nodes are distributed evenly across l layers, where l is variable. By this, shallow and deep NNs can be compared since they have the same complexity.

- **Activation function** For the present regression problem, suitable activation functions are the hyperbolic tangent or the ReLU, as they are used widely in this field. To find out, which one is preferable in this case, they will be also optimized in the grid search.

- **Weight initialization** The weights for the NN are initialized by the Xavier weight initialization procedure, which is the standard method used for NNs. By this, the variances of the activations are the same across all layers, preventing exploding or vanishing gradients. In this case, the normal Xavier initialization is applied as described in [41]. The gain factor is chosen according to the recommended value for the used activation function, thus either $\sqrt{2}$ for ReLU or $5/3$ for Tanh.

- **Number of epochs** The number of epochs can be estimated by the point where the validation loss increases while the training loss still decreases. It can be obtained simply from plotting training and validation loss of the NN.

- **Batch size** The optimal batch size is dependent on the problem, therefore different values are tested for it in the grid search. They are in the range of 100 to the full length of the training set.

- **Learning rate** Also the learning rate is optimized in the grid search. The tested values range from 10^{-1} to 10^{-4}. These ranges are typical values for the learning rate.

- **Loss function** The loss function for the training process is chosen to be the MSE because this is the generally used metric for regression problems.

- **Optimization algorithm** As the optimization algorithm, the Adam optimizer is used. First tests indicated that this works well here. However, if the performance of the training is insufficient, also other alternatives are tested.

The NN model is implemented using PyTorch [55]. For this, a NN class was built that includes methods to define the transformations for in- and outputs, the construction of the NN, and the training of the NN. All necessary hyperparameters can be handed over to these functions and thus constructing and training the NN breaks down to defining parameters and calling functions.
As mentioned before, the values of some hyperparameters are specific to the problem, thus they have to be inferred in an optimization procedure. This relates again to the

learning rate	batch size	# layer	activation
$10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}$	100, 1000, 5000, 20000	1, 4, 8, 16	ReLU, Tanh

TABLE 5.3: Tested hyperparameter values for the NN training.

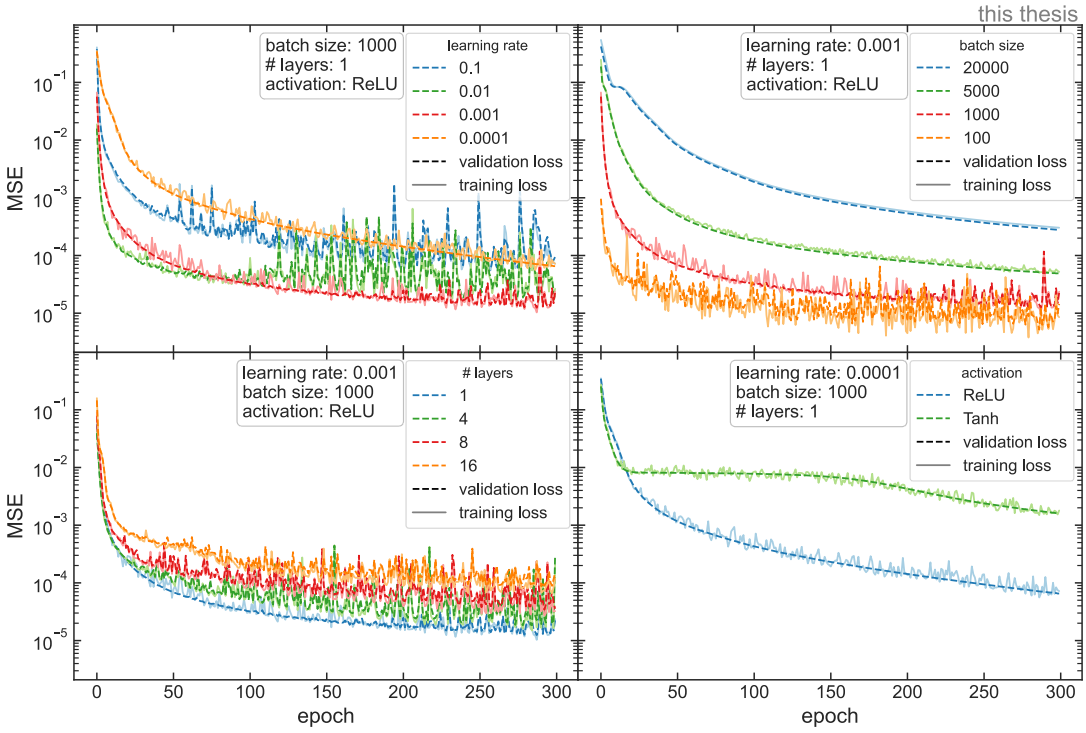


FIGURE 5.14: Effects of various NN hyperparameters on the training and validation losses during the training process.

broad topic of optimization of parameters, however, in this case, the hyperparameters are obtained with rather simplistic approaches, namely testing by hand and a simple grid search. But also more advanced methods like Bayesian parameter estimation could be used for it. To compare different architectures of NNs, the loss curves for the training and validation set can be considered. They monitor the MSE of the training and validation data after each epoch during the training process of the NN and thus provide a measure of performance and convergence of the NN. With that, conclusions about many of the parameters can be drawn. To examine the effect of the parameters on the loss curves, a grid search can be performed. Here, the grid includes the number of layers and nodes per layer, the activation function, the batch size, and the learning rate. The tested values for each of the considered parameters are given in tab. 5.3.
The total number of parameter combinations is $4 \times 4 \times 4 \times 2 = 128$ since the number of layers and the number of nodes per layer are varied in dependence on each other (# nodes/layers = 1024/# layers) to compare networks of the same complexity. The grid was generated and the hyperparameters of 128 NNs were set according to the grid points and fixed parameters. Then all NNs were trained on the batch farm at GSI and training and validation loss curves were stored. The loss curves of some NNs are shown in fig. 5.14.
In the figure, in each plot only one parameter is varied at a time, all other parameters were fixed to illustrate the effect specific parameters have. This corresponds to considering a 1D slice of the search grid. All effects that are visible, were also apparent for

other fixed parameters so it can be assumed that they are general.
At first, it is reasonable to discuss some general properties of the loss curves. It is visible, that the initial loss quickly decreases until it converges to some specific value for all loss curves. The shape of this decrease already provides a lot of information. Its steepness characterizes the convergence rate of the network, and steeper loss curves indicate that the network learns faster. The value the training converges to is a measure of the performance of the network, a lower MSE is connected to a better fitting NN. Also, the optimal number of epochs can be obtained by the loss curves. If the loss curves do not converge, it can be chosen larger, otherwise, the network can be trained until the loss has converged. In this case, the MSE has converged for most of the NNs, but for some, the number of epochs may even be increased. Another striking feature of all losses in fig. 5.14 is, that no overfitting occurs, which would result in diverging training and validation loss. The reason for this is that the used simulation is deterministic. Hence, there is no noise in the data that could be fitted, and fitting the training data better corresponds to fitting the model better.
In the upper left corner of fig. 5.14, the effect of the learning rate on the training process is depicted. For too small learning rates the training converges slower because the optimization algorithm only makes small steps in the gradient descent. For too large learning rates the validation losses start to fluctuate heavily because the optimization algorithm overshoots the minimum which also worsens the performance. Here, the optimal learning rate is 0.001.
In the upper right plot, the effect of different batch sizes is visible. Smaller batch sizes clearly benefit the training as they improve convergence and lead to a lower MSE. However, they also lead to larger fluctuations of the training loss because only a fraction of the data set is used to update the weights in each optimization step. In this case, a batch size of 100 is optimal since the converged MSE is also with its fluctuations lower than the other tested values.
In the lower-left corner, the network architecture was changed. The NNs with layers 1, 4, 8, and 16 correspond hereby with a number of nodes-per-layer of 1024, 256, 128, and 64, such that the total number of nodes in each NN is 1024. It is apparent, that shallow NNs perform better than deep NNs for the present problem. This may be the case because deeper networks are more susceptible to vanishing gradients such that the weights are learned less efficiently. For this work, a NN with one hidden layer will therefore be used.
The last hyperparameter that was tested is the activation function, which is depicted in the lower-right corner of fig. 5.14. For the generated parameter grid in all cases, the ReLU activation function was superior, therefore this will be used for the NN emulator model.
From the hyperparameter grid, the best-performing NN can be extracted by comparing the final losses of the validation data for all NNs. The minimally observed MSE is $5.4 \cdot 10^{-6}$ when using a learning rate of 10^{-4}, a batch size of 100, one hidden layer, and the ReLU activation function. Therefore, this network design is used for the emulator model. The loss curves for this configuration are shown in fig. 5.15.
The performed hyperparameter search is rather simplistic and only four of the hyperparameters were optimized. Furthermore, for the batch size and the learning rate, the optimal values are at the edge of the grid suggesting that there might be even better values outside of the search ranges. Hence, with a larger grid or more sophisticated optimization schemes, better hyperparameters could be found which increases the performance of the emulator model. However, it is expected that the chosen network is already performing sufficiently well and that further optimization leads only to a marginal performance boost. Thus this is omitted here.

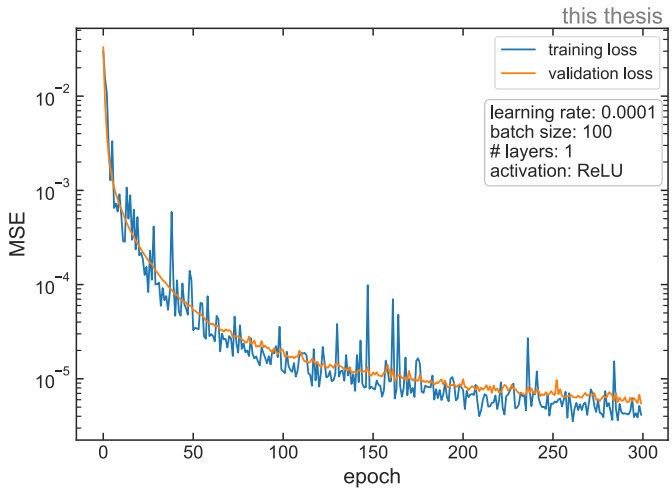


FIGURE 5.15: Training and validation losses of the best performing NN in the hyperparameter grid. This network structure forms the basis of the emulator model.

Performance of the NN

To quantify the performance of the trained NN, its output can be compared to the validation output. The MSE already provides an estimate of the performance, however, since it is based on the normalized outputs its absolute value is not informative. Therefore, other metrics are used here to examine the performance of the NN. One of them is the error of the NN prediction, which is given by the NN output minus the validation output of the simulation ($y_i^{\mathrm{NN}} - y_i^{\mathrm{model}}$). The error can then be compared to the experimental uncertainty of the output to get an impression of its magnitude. In fig. 5.16, the correlation of the NN predicted spectra output and the model output is shown for three different p_{T}-bins for three different particles for three centrality classes as well as the ratio between the prediction error and the data uncertainty for the same bins. The neural network predictions are strongly correlated to the model outputs, which indicates that the NN emulates the simulation well. It does this not only for the data it was trained on but also for the validation data. On the right side of the figure, the distributions for the ratio between the prediction error and the data uncertainty for the three cases are plotted. The error is distributed around zero in a Gaussian-like shape and the distributions for training and validation data are approximately the same. In all bins, the prediction error is much smaller than the experimental data uncertainty, thus the Bayesian inference will be mostly driven by the experimental uncertainty and further improvements to the NN architecture will not affect the result of it much.
In fig. 5.16, only three of the 500 considered p_{T}-bins are illustrated, but the correlations and distributions for all of them were checked. No deviation from depicted behavior was found. The combined distribution of all 500 p_{T}-bins is depicted in fig. 5.17. The prediction error rarely exceeds 30 % of the experimental uncertainty. The largest observed value for the ratio is 3.57 which is an outlier.
Having defined and trained the NN, a fast surrogate model for the simulation is available. To illustrate its outputs and deviations to the experimental data, it is convenient to consider the χ^2-value as this is related to the calculation of the Bayesian posterior probability and reduces the information of the 500 spectra values to just one value. In fig. 5.18, two-dimensional slices of the search space are visible, in which the χ^2-value between emulator and data is encoded in the color. The plots were generated by applying the NN to parameter grids and then the χ^2 was calculated using the experimental

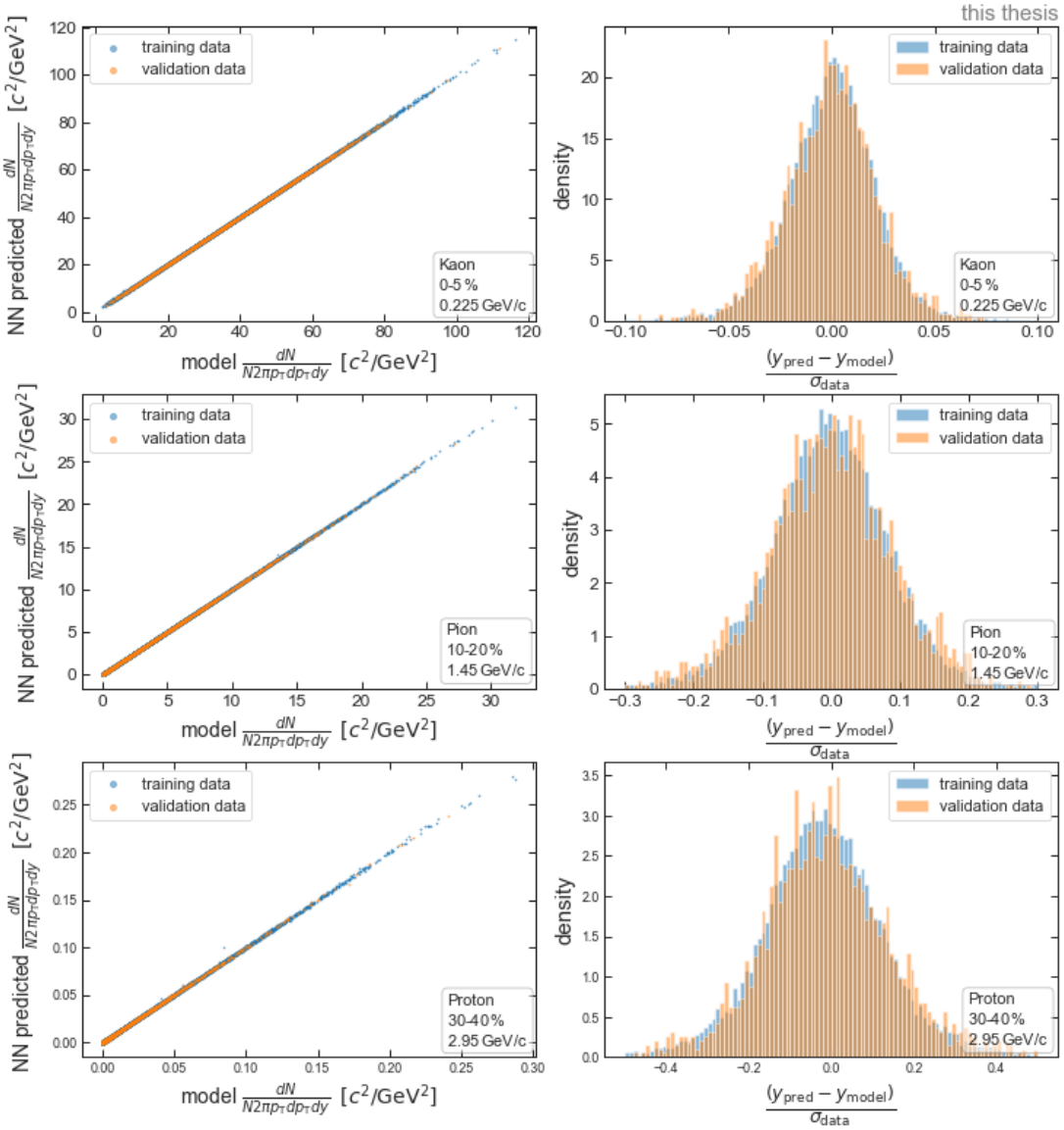


FIGURE 5.16: Left: Correlation between the NN prediction and the simulation output for training and validation data. The NN clearly reproduces the simulation data. Right: Prediction error in units of the experimental error. The prediction error is well below the experimental uncertainty and distributed around zero.

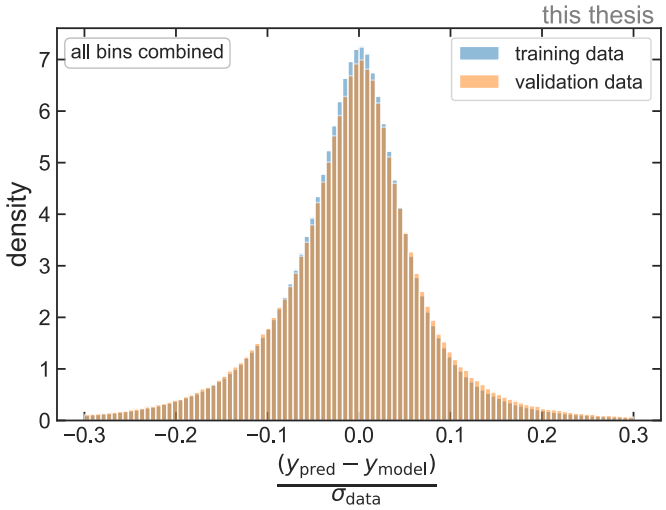


FIGURE 5.17: Distribution of the prediction error in units of the experimental uncertainty for the validation and training data for all p_{T}-bins combined.

data. In these plots, regions of lower χ^2, respectively better fits to data, are visible, which an optimization procedure would converge to. However, the produced plots are only presented here for illustration purposes, the apparent minima most certainly are not the global minimum, since only slices of the six-dimensional parameter space are considered.

5.5.3 NN ensemble emulator model

Having defined and trained the NN model, an ensemble of NNs can be built. The ensemble model is chosen rather than a single NN as an emulator to obtain also model uncertainty estimates for each prediction. Thus, the prediction error does not have to be calculated by considering the deviation from the simulation outputs. In this work, the ensemble model will be built by averaging the results of several NN and using the standard deviation of their prediction as an estimate of the model uncertainty. This approach was already introduced in sec. 4.2.1, the corresponding formulas for the calculation of the ensemble prediction and the prediction uncertainty are given there in eqs. 4.7 and 4.8. This simple approach is mainly used because only the model uncertainty but no data uncertainty has to be considered here due to the deterministic nature of the simulation. For problems where also data uncertainty is prevalent, more sophisticated methods like deep ensembles [46] may be used.
For the ensemble, two requirements must be met: Each ensemble member has to be as good as possible, and the individual members have to be as diverse as possible. The first point was already addressed in the last section, so what remains is the introduction of diversity into the ensemble. As was argued in sec. 4.2.1, the ensemble members are sufficiently different if random weight initialization and random shuffling are used. Hence, these two methods will be the only ones that are considered to induce variety. From a computational standpoint, this is not connected to much further effort since these techniques are already implemented in the network presented in the last section. To build the ensemble, therefore, only several NNs have to be trained and then their predictions can be combined to a mean ensemble prediction with a standard deviation. Until now, it has not been discussed, what the optimal number of NNs in an ensemble is. To examine this, the performance of ensembles with a varying number of members

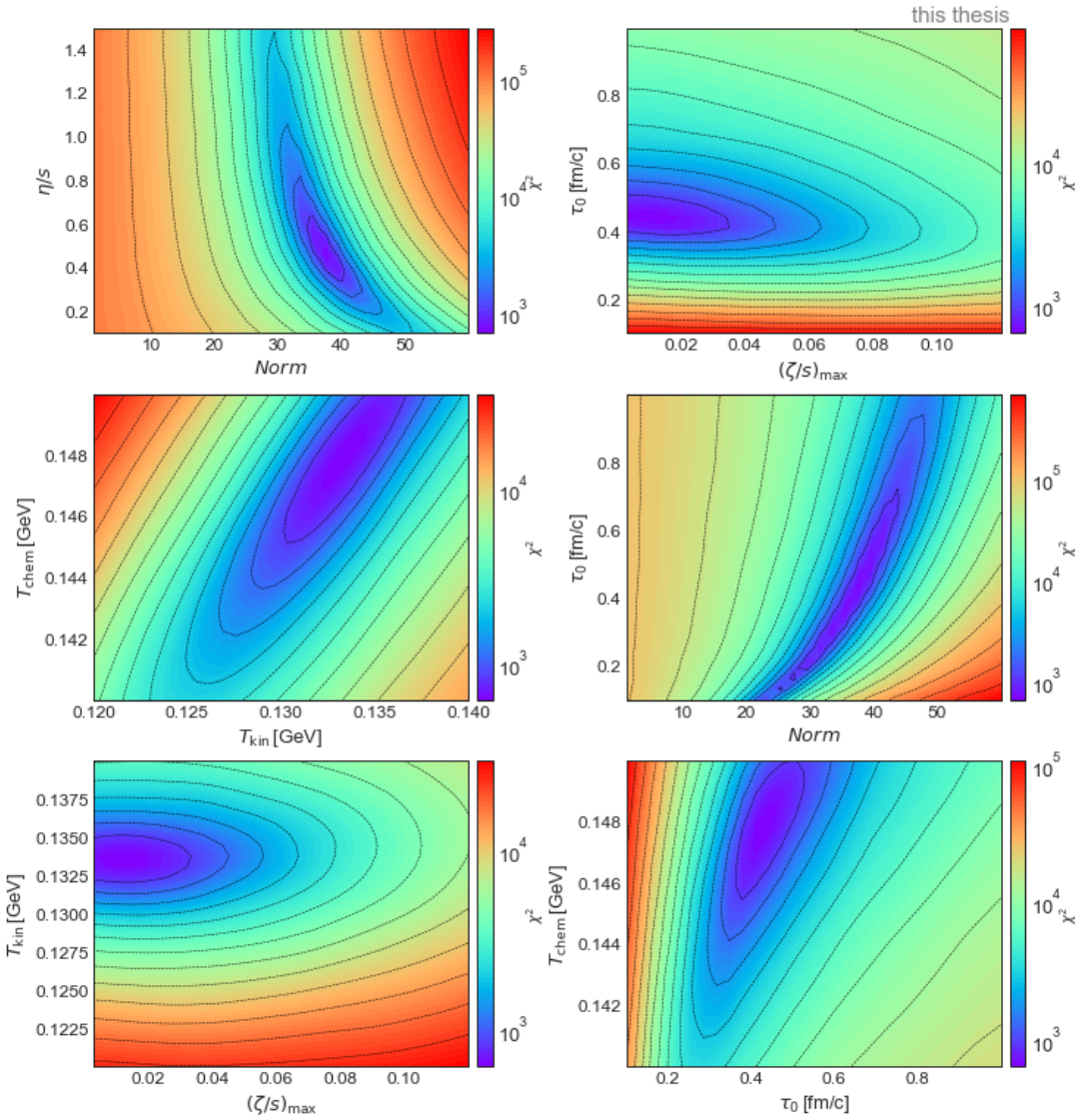


FIGURE 5.18: χ^2-values across slices of the parameter grid generated with the NN. Clearly visible are regions with larger and lower χ^2, which indicate worse and better parameters of the model.

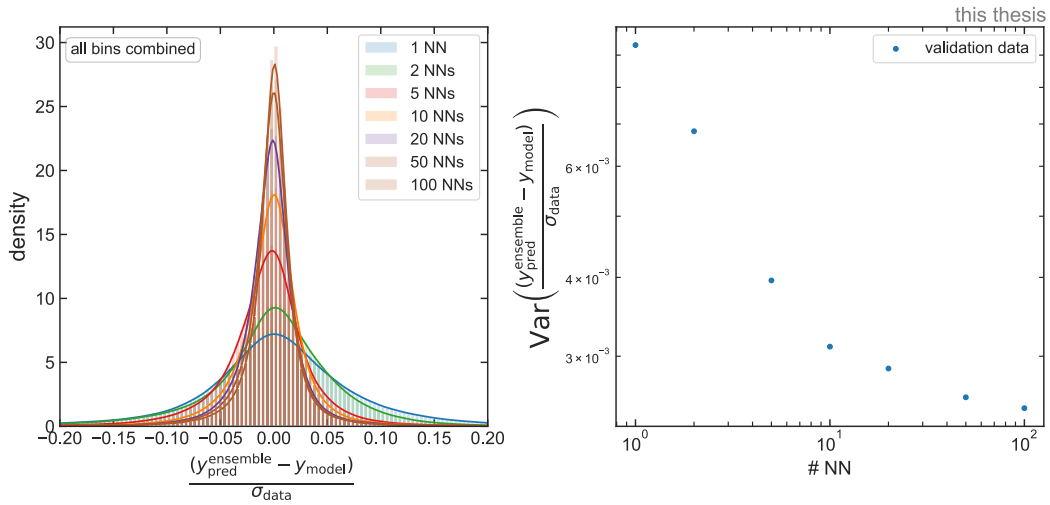


FIGURE 5.19: Left: Prediction error in units of data uncertainty for an ensemble with different numbers of NNs. Right: Decrease of the variance of the prediction for an increasing number of NNs.

can be evaluated. To construct the ensembles, 100 NNs are trained with the optimized hyperparameters found in the last section. Each NN is trained with random initialization and data shuffling, which ensures diversity among the NNs. Then, 7 ensembles with 1, 2, 5, 10, 20, 50, and 100 members are constructed from the 100 NNs to test their performance.

At first, the prediction error is considered, which is the ensemble output prediction (average prediction of all members) minus the true output obtained from the simulation ($y_{\text{pred}}^{\text{emsemble}} - y_{\text{model}}$). This error is calculated for all outputs of the validation dataset, including all p_{T}-bins of all particles in all centrality classes. To get a feeling for the size of this error, it can be normalized again by the corresponding experimental uncertainty for the specific p_{T}-bin. The distribution of the prediction error over the experimental uncertainty for all p_{T}-bins of the validation set combined is shown for the 7 ensembles in fig. 5.19.

It is apparent, that the prediction errors decrease for ensembles with an increasing number of NN. This is expected since the ensemble prediction, an average of its members, gets more precise for larger sample sizes. On the right side of fig. 5.19, the variance of the distributions of the left side is depicted to quantify the increase in accuracy. The variance drops significantly by a factor of ~ 3 from 1 NN to 10 NNs and then from 10 to 100 by another factor of ~ 1.2. The observed behavior does not follow the $1/\sqrt{N}$ law, which is the expected functional form for the average of an uncorrelated sample and would result in a linear function in the log-log plot. The reason for this is correlations between the NNs. Each network is trained on the same data and has the same hyperparameters, thus they tend to deviate from the true value in the same direction. Hence, adding new correlated NNs to an ensemble adds less information than adding uncorrelated ones and the variance decreases less pronounced. The correlation is discussed later in more detail again.

The results of fig. 5.19 imply that the ensemble for the parameter estimation should be chosen as large as possible to minimize the prediction error. But increasing the number of members of an ensemble is also associated with an increased computational effort for training and evaluation, such that too large ensembles are impractical to use. As a compromise between accuracy and computational performance, in the following, an ensemble with 10 NNs is selected for the emulator model. For 10 NNs, the ensemble

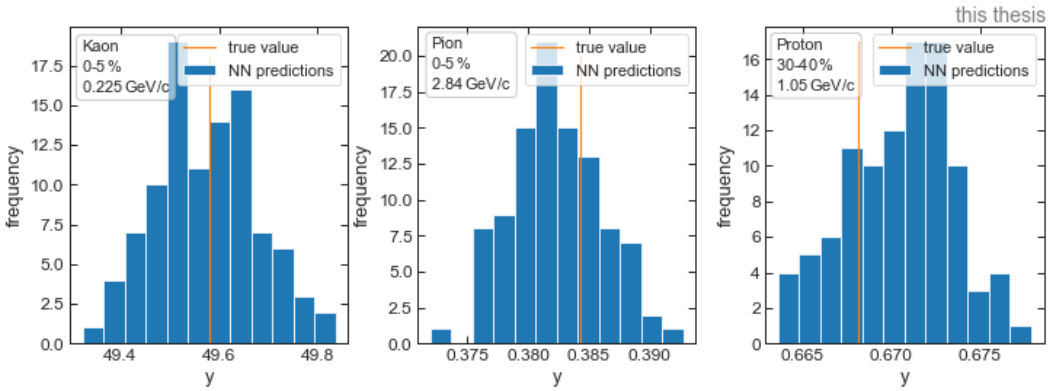


FIGURE 5.20: Examples of the NN predictions for three p_T-bins and random input parameters taken from the validation set.

variance is reduced significantly by a factor of ~ 3 compared to a single NN, and the computational effort is still manageable. Any further NN contributes only little to the overall performance, thus adding more members is not worth the additional computational effort.

Until now, only the mean prediction of the ensemble has been considered. However, the ensemble has been introduced primarily to account for the model uncertainty of the NNs, which also has to be investigated. A priori it is not clear if the difference in the predictions of individual NNs can describe the model uncertainty, although previous results support this claim [46]. To investigate this, at first, the spread of the predictions of the NNs in the ensemble is considered. In fig. 5.20, the predictions of the ensemble members are shown for three different input parameter configurations from the validation set together with the true output from the simulation. For all three cases, the predictions are distributed in a Gaussian-like shape, not necessarily around the true value, but the distribution always includes it. To check if the apparent samples can be identified with a Gaussian, the Shapiro-Wilk test [56] can be performed. This test tests the hypothesis that a sample comes from a Gaussian distribution. The significance level is set to 0.05 for the test. The p-values for the three different cases of fig. 5.20 are 0.55, 0.21, and 0.57, which are larger than the significance level. Therefore, the null hypothesis can not be rejected and the samples can be assumed to come from a Gaussian distribution. This test can be carried out for all 1954×500 outputs of the validation data set to ensure that the Gaussian assumption is true for all regions in the parameter space and for all p_T-bins in all centrality classes and for all particles. From the 1954×500 tests, the hypothesis could be rejected in 48,918 cases, which is a fraction of 0.05007. This value is the expected value of rejected hypotheses if the null hypothesis is true for a significance level of 0.05. Therefore, the distribution of the NN predictions can be assumed everywhere to come from a normal distribution.

What remains is the calculation of the uncertainty of the ensemble prediction. The commonly used procedure is to estimate it simply by the standard deviation of the predictions of the NNs in the ensemble, which was used for example in [46]. Because of two reasons, this is not accurate: Firstly, the prediction of the ensemble will be the mean prediction of all NNs in the ensemble. Thus the error *of the mean* has to be considered rather than the error of a single NN. And secondly, any correlation between the networks is not considered, which heavily impacts the estimate of uncertainty.

To include both effects in the uncertainty estimate, let's assume the NNs predictions to be samples of random variables X_i for fixed input parameters. Then, in the general case of correlated random variables, e.g. correlated NNs, the variance of the sum can

be obtained by

$$\mathrm{Var}\left(\sum_{i=1}^{N} X_i\right) = \sum_{i=1}^{N}\sum_{j=1}^{N} \mathrm{Cov}\left(X_i, X_j\right) = \sum_{i=1}^{N} \mathrm{Var}\left(X_i\right) + 2 \sum_{1\leq i<j\leq N} \mathrm{Cov}\left(X_i, X_j\right), \quad (5.14)$$

where N is the number of variables. This is a sum over all entries of the covariance matrix. Dividing by $1/N^2$ then gives the variance of the mean. Eq. 5.14 can be simplified for the case that all variables have the same variance σ^2, which yields

$$\mathrm{Var}(\bar{X}) = \frac{\sigma^2}{N} + \frac{N-1}{N}\rho\sigma^2, \quad (5.15)$$

where $\bar{X}$ is the average prediction, and ρ is the average correlation between two variables. σ^2 in the case of the NNs is the uncertainty of a single NN $\sigma^{\mathrm{NN}}_{\mathrm{pred}}$. The assumption that this is the same for all NNs is reasonable since all NNs in the ensemble are trained in the exact same way with the same data, thus their predictions will exhibit the same magnitude of uncertainty. The uncertainty $\sigma^{\mathrm{NN}}_{\mathrm{pred}}$ is not accessible directly either but may be estimated by the spread of the NN predictions. It is given by

$$\sigma^{\mathrm{NN}}_{\mathrm{pred}} = \frac{1}{\sqrt{1-\rho}}\hat{\sigma}^{\mathrm{NN}}_{\mathrm{pred}}, \quad (5.16)$$

where $\hat{\sigma}^{\mathrm{NN}}_{\mathrm{pred}}$ is the standard deviation of the NN predictions and ρ is again the mean correlation. The factor $\sqrt{1-\rho}$ accounts for the correlation between the NNs. With this, the full expression of the uncertainty estimate of the mean is

$$\sigma^{\mathrm{ensemble}}_{\mathrm{pred}} = \sqrt{\frac{\frac{1}{N} + \frac{N-1}{N}\rho}{1-\rho}}\hat{\sigma}^{\mathrm{NN}}_{\mathrm{pred}}, \quad (5.17)$$

which implies that the correlation effectively accounts for a correction factor of the standard deviation of the NN predictions. If no correlation is apparent, the formula reproduces the error of the mean for uncorrelated samples $1/\sqrt{N}\hat{\sigma}^{\mathrm{NN}}_{\mathrm{pred}}$. Thus, the ensemble prediction uncertainty can be calculated by multiplying the correction factor with the standard deviation of the spread of the NNs in the ensemble. However, the mean correlation ρ, and therefore the correction factor, is unknown a priori, so it has to be estimated from the data itself. To do so, the distribution of the error of the ensemble prediction $(y^{\mathrm{ensemble}}_{\mathrm{pred}} - y_{\mathrm{true}})/\sigma_{\mathrm{pred}}$ can be considered. This is given by a standard normal distribution if the prediction uncertainty σ_{pred} captures the prediction error. σ_{pred} can be expressed by $c\sigma^{\mathrm{NN}}_{\mathrm{pred}}$, where c is chosen according to eq. 5.17. Because of this, the distribution $(y^{\mathrm{ensemble}}_{\mathrm{pred}} - y_{\mathrm{true}})/\sigma^{\mathrm{NN}}_{\mathrm{pred}}$ has a standard deviation of c. The correction factor may therefore be obtained by fitting this distribution.
Nevertheless, additionally, it has to be taken into account that the uncertainty $\sigma^{\mathrm{NN}}_{\mathrm{pred}}$ has to be estimated by the sample standard deviation $\hat{\sigma}^{\mathrm{NN}}_{\mathrm{pred}}$ itself since the true value is unknown. Therefore, the ratio $(y^{\mathrm{ensemble}}_{\mathrm{pred}})/\hat{\sigma}^{\mathrm{NN}}_{\mathrm{pred}}$ will not be normally distributed but will follow a Student's t-distribution, which is similar to a normal distribution but exhibits heavier tails. It is defined by the number of degrees of freedom, given by $N-1$, where N is the number of NNs in the ensemble. For large N, the t-distribution converges to the normal distribution, but in the present case with an ensemble size of 10 NNs, there is still a significant difference to it.

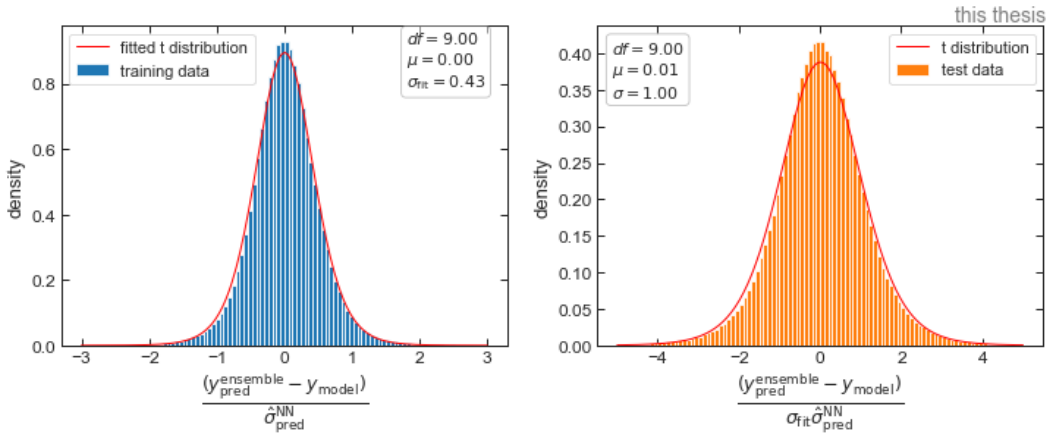


FIGURE 5.21: Left: Determination of the correction factor of the ensemble uncertainty by fitting a t-distribution. Right: Corrected distribution for the test data set.

Taking into account all these considerations, the correction factor for the uncertainty of the ensemble prediction can be estimated by fitting a t-distribution to the distribution $(y_{\text{pred}}^{\text{ensemble}})/\hat{\sigma}_{\text{pred}}^{\text{NN}}$. In principle, this distribution has to be constructed and fitted for every input parameter configuration and every p_{T}-bin, which is infeasible since the true model values are only known at the training or validation points. To simplify the procedure, it is assumed that the correlation, and therefore the correction factor, are independent of the position in the parameter space and the considered p_{T}-bin, which allows collecting all errors from all p_{T}-bins and points in all regions of the parameter space in one distribution. The assumption holds because the density of points in all regions of the parameter space is the same and all p_{T}-bins are treated in the same way, which leads to similar correlations between the networks. This assumption is, however, not strictly true because the density varies on a small scale, so some deviations may occur. If the assumption does not hold this will be visible in the distribution, as it is then described by a sum of t-distributions instead of a single one.
In fig. 5.21 on the left side, the distribution $(y_{\text{pred}}^{\text{ensemble}})/\hat{\sigma}_{\text{pred}}^{\text{NN}}$ is shown for the training data, combined for all p_{T}-bins. The figure also shows a fitted Student's t-distribution. It has 9 degrees of freedom $(\#NN - 1)$ and the correction factor, e.g. the scale of the distribution, is estimated to be $\sigma_{\text{fit}} = 0.43$.
From fig. 5.21 it is obvious, that the t-distribution describes the form of the data well thus the assumption of the constant correlation is reasonable. Only a slight deviation compared to the distribution is visible. The correction factor $\sigma_{\text{fit}} = 0.43$ is equivalent to a mean correlation of $\rho = 0.0806$ according to eq. 5.15. In the right plot of fig. 5.21, the corrected distribution for the test data is depicted. The corrected uncertainty is able to describe the ensembles' prediction error as the expected t-distribution with 9 degrees of freedom and a scale of one is approximately retained.
To cross check if eq. 5.17 is describing the uncertainty correctly, the scaling factors for the distributions of ensembles with different numbers of members can be computed by fitting t-distributions. As it can be expected that the average correlation between two networks is the same regardless of the size of the ensemble, the correction factors should follow the functional form of eq. 5.17 in dependence of N with $\rho = 0.0806$, approximately. In fig. 5.22 on the left, the error distributions are plotted for ensembles with 2, 5, 10, 20, 50, and 100 NNs. To these distributions, t-distributions were fitted, and the correction factor was extracted and plotted on the right side in dependence on the number of NN in the ensemble. The arising points were then fitted with the expected

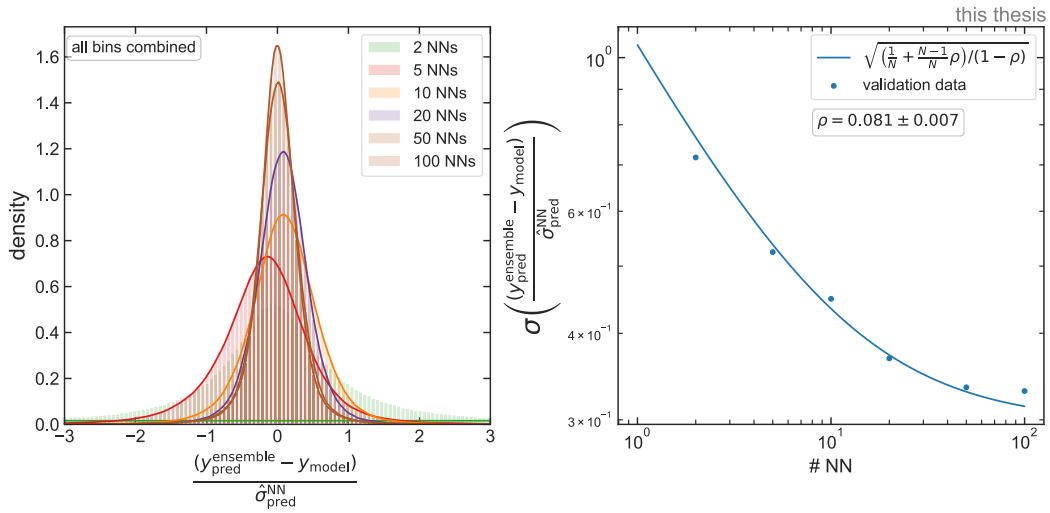


FIGURE 5.22: Left: Decrease of the prediction error due to averaging over the ensemble members. Right: Variances of the distributions on the left in dependence of N and the expected functional form.

functional form.
It is apparent, that eq. 5.17 is able to describe the general dependence between the correction factor and the number of NNs in the ensemble. The estimated correlation is $\rho = 0.081 \pm 0.007$, which agrees with the previously determined value of $\rho = 0.0806$.
As mentioned before, the reason for the introduction of the emulator model is to efficiently obtain outputs for given inputs mimicking the simulation. In the previous sections, it has been shown and verified that the ensemble accurately reproduces the simulation, hence, what is left to be discussed is the decrease in computation time between simulation and emulator. To examine this, the simulation and the ensemble can be evaluated for a set of input parameters and the running times can be compared. While the simulation takes about 3 minutes to compute the output for one set of parameters, the ensemble takes only 0.01 seconds, reducing the computation time by a factor of 10^5. The large decrease makes the MCMC simulation applicable in the first place. The computational performance of the emulator is even higher if batches of input parameters are provided to the ensemble because of vectorization. For example, 1000 configurations computed in parallel take about 0.6 s. Furthermore, the ensemble model is easily parallelizable on a computing cluster as it is coded in python, whereas the parallelization of the simulation is limited by the number of available Mathematica licenses on the cluster.
In conclusion, the constructed ensemble model provides an efficient and accurate emulator model to obtain the outputs of the simulation for given inputs. Moreover, any deviation between the emulator model and the simulation is quantified by it, thus providing well-calibrated uncertainty estimates of its predictions. This model may now be used to replace the simulation.

5.6 MCMC simulation

After an efficient emulator model for the simulation has been constructed and verified, it can be used for the ultimate goal of this thesis, the determination of the posterior probability distributions using Bayesian inference. For that, the posterior probability is calculated by eq. 5.6, effectively multiplying the prior probability chosen by eq. 5.8 with the likelihood given in eq. 5.12. As already introduced in sec. 4.3, the probability

space is most efficiently populated by an MCMC simulation. Computationally, this will be taken care of by the emcee python package [57], which is an implementation of the affine-invariant ensemble sampler proposed by Goodman and Weare [58]. For running the MCMC algorithm, the logarithmic probability of the considered distribution has to be quantified. In the present case, this is given by log of the posterior probability:

$$\log(P(D|\mathbf{x})) \propto \begin{cases} -\frac{1}{2}\left[\mathbf{y}_m(\mathbf{x}) - \mathbf{y}_e\right]^{\mathrm{T}} \Sigma^{-1} \left[\mathbf{y}_m(\mathbf{x}) - \mathbf{y}_e\right] & \text{if } x_i^{\min} \leq x_i \leq x_i^{\max} \,\forall\, i \\ -\infty & \text{else} \end{cases}, \tag{5.18}$$

where x_i, $i = 1, 2, ..., 6$ are the input parameters, $x_i^{\min}$ and $x_i^{\max}$ are their limits, $\mathbf{y}_e$ is a vector of the experimental spectra for all considered particles and centralities, $\mathbf{y}_m(\mathbf{x})$ is the according vector for the emulator model for inputs $\mathbf{x}$, and Σ is the covariance matrix consisting of the data and model covariance matrices ($\Sigma = \Sigma_e + \Sigma_m$). The experimental data $\mathbf{y}_e$ was already specified in sec. 5.1 and the emulator output $\mathbf{y}_m(\mathbf{x})$ is straightforward to compute, thus only the covariance matrices have to be discussed here. For the experimental data, only the statistical and systematic uncertainties are available for each spectra value, but no information about their correlations. Therefore, the data covariance matrix consists of entries only on the diagonal. These elements are given by the squared sum of the statistical and systematic uncertainties ($\sigma^2 = \sigma_{\text{stat}}^2 + \sigma_{\text{sys}}^2$). Omitting correlations results in larger posterior probabilities and lower χ^2-values, which impacts the final posterior distributions, however, since no information is available, including them is infeasible at the moment.
The second covariance matrix that is considered is the emulator model covariance. This covariance can be simply computed with the output of the NNs predictions in the emulator for each input configuration. The NNs provide a matrix of size $N \times 500$, where N is the number of NNs. The covariance is then given by

$$\mathrm{Cov}(y_j, y_k) = \frac{1}{N-1} \sum_{i=1}^{N} \left(y_{ij} - \bar{y}_j\right)\left(y_{ik} - \bar{y}_k\right), \tag{5.19}$$

where y_j is the jth spectra output value and $\bar{y}_j$ is the mean prediction for the jth output. To obtain the covariance matrix of the mean prediction, all entries have to be scaled by the correction factor of 0.43^2. The covariance matrix can be calculated by this procedure for each prediction. To get an impression of the covariance structure, the correlation matrix, which is connected to the covariance matrix, is depicted in a heatmap in fig. 5.23.
What is shown is the average correlation of the spectra values from the spectra of Pions, Kaons, and Protons in all considered five centrality classes. The matrix consists of 500×500 correlations between the p_{T}-bins. The particle and centrality classes are displayed on the axes, but not the specific p_{T}-values because of visualization purposes. The correlation of the spectra of two particles and centrality classes is depicted in the box substructure. From the boxes of the main diagonal, the correlation of neighboring p_{T}-bins in the spectra can be extracted. As expected, these are highly correlated, as the ensemble produces a continuous output. In contrast to this, p_{T}-bins of the same spectrum far apart from each other are weakly correlated. Because of these two effects, the correlation between spectra exhibits a strong diagonal structure, which is visible in the figure. Additional to the main diagonal there are several parallel diagonals that are apart by three boxes. These account for the correlations between different centrality classes of the same particles, which also contain strong correlations. Furthermore, pions and kaons of the same centrality classes have strong correlations, protons, however, are less correlated with these particles.

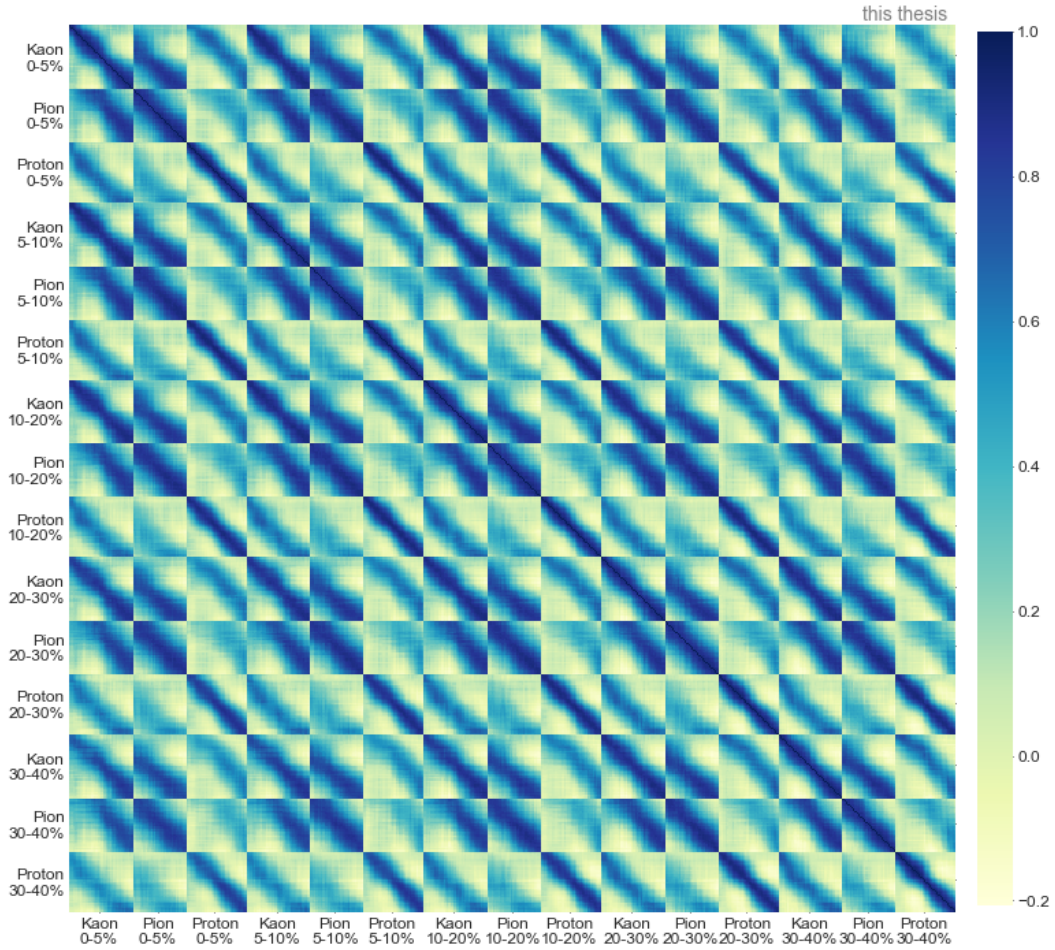


FIGURE 5.23: Average correlation between the outputs of the emulator for the validation data.

Having defined all components of eq. 5.18, the MCMC simulation can be deployed. For this, two additional parameters have to be set, namely the number of walkers and the number of steps per chain. These are important because if chosen too small, the chains may not be converged to the equilibrium distribution, leading to distorted results. To ensure converged distributions, the needed length of the chains and the number of chains can be estimated using the integrated autocorrelation time τ_f. As explained in sec. 4.3, the sampling error of the MCMC method is given by $\sqrt{\tau_f / N}$ governed by an *effective* sample size N/τ_f, taking into account the autocorrelation in the chains. In this analysis, the distributions will be considered converged, if the sampling error is below 1 %. This implies, that the sample size of the chains has to be $N \geq 10000\tau_f$. It does not matter if this sample size is reached by running a lot of parallel chains with a small number of steps or running only one chain with a large number of steps due to the ergodicity of the Metropolis-Hastings sampler that is used. This means that the product of the number of chains times the length of the chains has to be greater than $10000\tau_f$ to reach an error below 1 %. The number of walkers is chosen to be 64 and the MCMC simulation will be stopped if the chains reach a length of $200\tau_f$, which leads to $12800\tau_f$ generated samples. The integrated autocorrelation time is approximated during the generation of the chain, every 500 steps it is calculated by the integrated autocorrelation time method of emcee. The estimate is unreliable if the chains are smaller than $50\tau_f$, as explained in [57], thus to prevent a too early stopping of the MCMC simulation a second

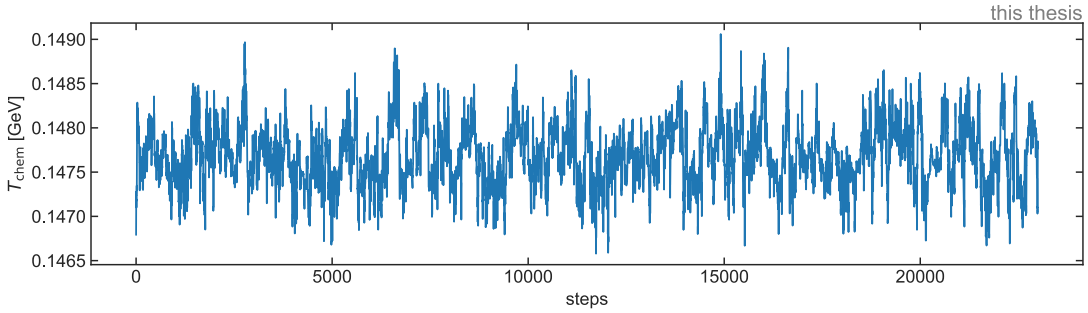


FIGURE 5.24: Example of an MCMC chain projected onto the T_{chem}-axis. On a small scale the autocorrelation is visible.

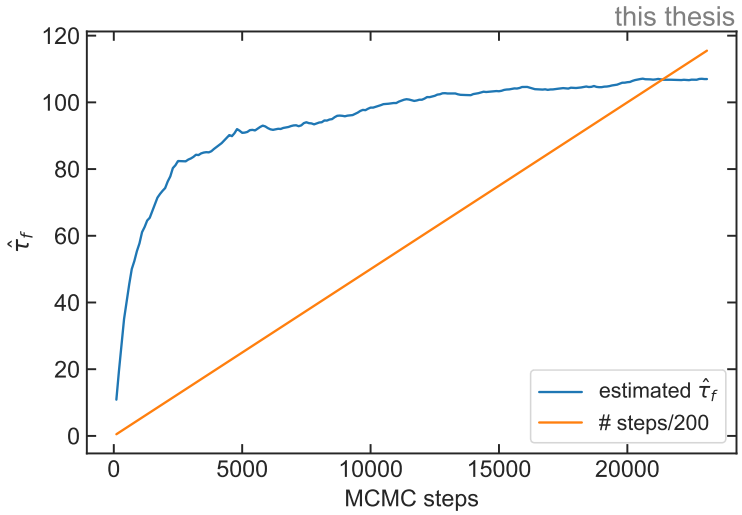


FIGURE 5.25: Estimate of the integrated autocorrelation time $\hat{\tau}_f$ in dependence of the length of the chain. For small lengths, the estimate is unreliable.

criterion has to be introduced, which ensures that the algorithm is only stopped if also the estimate of the integrated autocorrelation time has converged. This is assumed to be the case if the change of τ_f between two calculation steps (500 MCMC steps) is less than 1 %.

Following this procedure, an MCMC simulation was run using the emulator model. The generated output consisted of 64 chains of a length of 23000. In fig. 5.24, the output for one chain projected to one axis is depicted.

From the figure, it is apparent that on small scales there is autocorrelation between the samples. On the broad scale, however, the distribution looks like a stationary random process, indicating that the chain has converged to the equilibrium distribution. To ensure that both convergence criteria are met, the estimates of the integrated autocorrelation time $\hat{\tau}_f$ are plotted in dependence on the number of steps in fig. 5.25. Additionally, the stopping criterion is shown.

In principle, the integrated autocorrelation time is constant, which would result in a horizontal line in the plot. However, since its value is not known, it has to be estimated from the chain. For short chains, this estimate is poor and completely underestimates the true value. Only as the length of the chain increases, the estimate of τ_f becomes more reasonable and finally converges to the true value. The intersection with the linear function (# steps/ N) marks the point where the first stopping criterion is met. At that

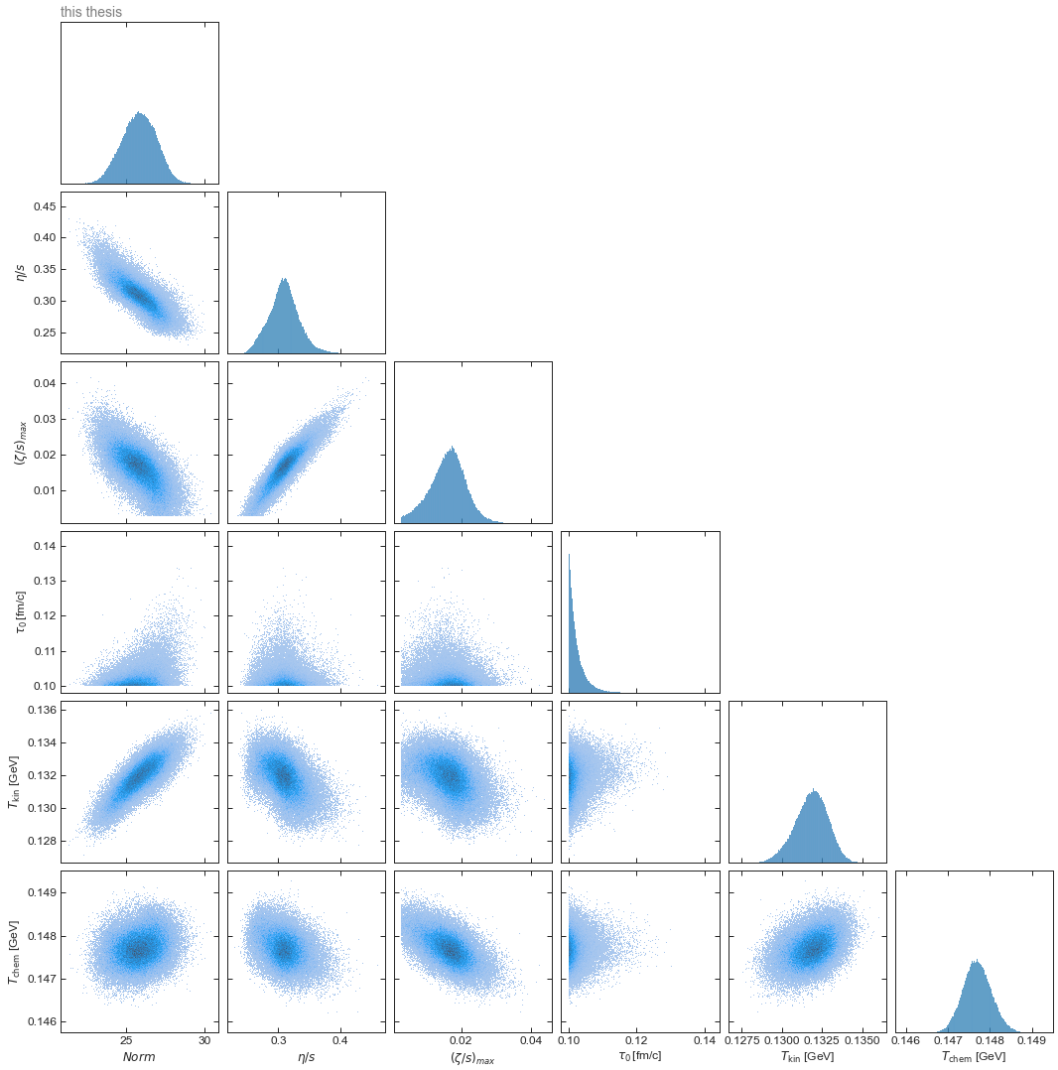


FIGURE 5.26: Marginal posterior probability distributions in the search space for all parameters.

point, the chain has a length of $200\hat{\tau}_f$. It is then prolonged until the estimate of $\hat{\tau}_f$ has converged, meaning that its change is less than 1 % between two calculation steps. The ultimate estimate of the mean integrated autocorrelation time is 103. The plots verify that the distribution has converged.

To infer the estimates for the input parameters, the marginal posterior probability distributions have to be constructed. Mathematically this is done by integrating out all but one parameter, numerically this can be achieved by projecting the six-dimensional distribution onto one axis. To also account for correlations of the parameters, the posterior distribution can also be projected into two-dimensional spaces with the considered parameters as axes. The arising marginal posterior distributions are shown in fig. 5.26. From the six input parameters, five, namely *Norm*, η/s, $(\zeta/s)_{\mathrm{max}}$, T_{kin}, and T_{chem} are well contained within the search ranges. Only the thermalization time τ_0 seems to be not captured by the search region as the MCMC simulation runs against the boundary at 0.1 fm/c. This may also impact the marginal distributions of all other parameters, thus their values have to be taken with care. It is also visible, that there is some correlation among the parameters. The shear viscosity η/s and the maximum bulk viscosity $(\zeta/s)_{\mathrm{max}}$ are strongly positively correlated, whereas both of them are negatively correlated with the *Norm* parameter. The *Norm* is also correlated with the kinetic freeze-out

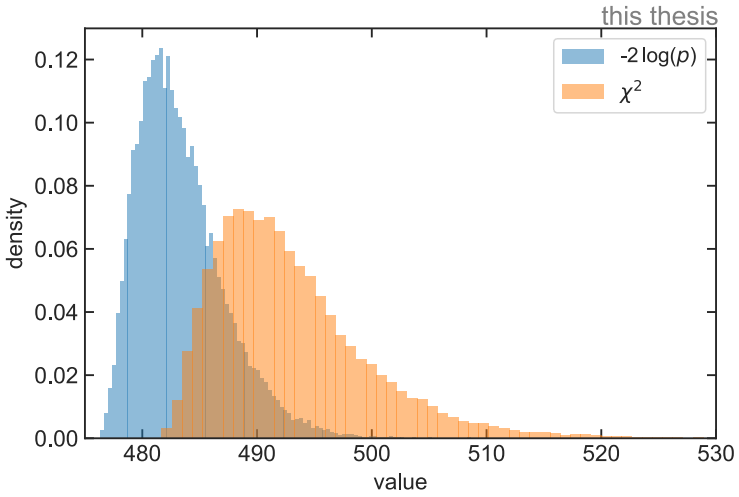


FIGURE 5.27: Log posterior probability and χ^2-values of visited parameter configurations in the MCMC simulation. Both are calculated with the NN ensemble. The difference is due to the consideration of the ensemble uncertainty in the $\log(p)$ calculation.

temperature T_{kin}. All other combinations exhibit no or small correlations.
Before the plots are discussed in more detail it is worthwhile having a closer look at the region where the MCMC simulation converged to. For this, the log posterior probability of the visited points in the MCMC simulation can be considered. This is closely related to the χ^2-value, for vanishing model uncertainty the χ^2-value is given by $-2\log p$. The distribution of the log posterior probability for the MCMC simulation is shown in the form $-2\log p$ in fig. 5.27. Additionally, to explore the effect of the ensemble uncertainty on the MCMC simulation, the distribution of the χ^2-values, omitting the ensemble uncertainty, is depicted.
From the figure two effects are apparent: Firstly, taking into account the ensemble uncertainty increases the posterior probability and has a significant impact on the distribution. And secondly, the MCMC simulation runs only in a very small region of the probability space with differences in the log probability of only 5 and maximum χ^2 differences in the order of 30. In comparison, the χ^2-values of the training samples range between a few hundred and a few million. Even the training sample with the smallest χ^2 of 639 is far above the region where the MCMC simulation converged to with a χ^2 of about 490. This means, that the region of highest probability is obtained fully by the interpolation of the NN ensemble. Additionally, the region is at the edge of the search grid at $\tau_0 = 0.1\,\text{fm/c}$, thus it is less constrained by the training samples. From both of these observations, it can be concluded that additional training points from the region of highest probability may be beneficial to further reduce the ensemble error and improve the posterior parameter estimates.

5.7 Refining the results

To refine the estimate of the parameters, the idea is to generate new points in the region of the highest probability to then train a new, more accurate, ensemble on these points which is then used to run another MCMC simulation to obtain the posterior estimates. This is essentially a second iteration of the introduced procedure. For this, the following steps have to be performed:

1. Generate uniformly new training and test points in the region of the highest probability of the first iteration.
2. Train and test the new ensemble model using the new data. Infer the correction factor.
3. Run an MCMC simulation using the new and improved ensemble model.

The first step can be done for example by using rejection sampling and uniformly sampling points in the search space and rejecting all that have a χ^2-value above some threshold. Or, as it will be done here, new points can be sampled from an MCMC simulation. For that, the following log probability may be used:

$$\log p = \begin{cases} \text{const} & \text{for } (\log p)_{\text{it0}} > \text{p}_{\text{threshold}} \\ -\infty & \text{else,} \end{cases} \tag{5.20}$$

where $(\log p)_{\text{it0}}$ is the log posterior probability defined in eq. 5.18, and $\text{p}_{\text{threshold}}$ is some chosen threshold that defines the region of highest probability. An MCMC simulation running with these settings uniformly generates samples from the region of highest probability. That the samples are distributed uniformly in the space is important as one assumption of the ensemble method is, that the correlation is constant with respect to the position in the parameter space, which is only approximately given if the density of points is constant.
For the present analysis, the threshold $\text{p}_{\text{threshold}}$ was set to -280. This is well containing the region of the MCMC simulation which runs at log probabilities of -250 to -240 and additionally includes points around the region of highest probability to better constrain the NN ensemble. The MCMC simulation was run and from the chains, points were sampled by calculating the integrated autocorrelation $\hat{\tau}_\text{f}$ time and taking every $\hat{\tau}_\text{f}$th point of the chain. This is done because the preceding points of the Markov chains are not independent of each other. The procedure resulted in 7177 parameter configurations, uniformly distributed in the region of highest probability. These were then divided into a training and a test set (test size: 0.2) and fed into the simulation to obtain new training and test points for the ensemble.
Having defined a new training and test set, an ensemble model can be constructed again following the procedure explained in the previous sections. As this was already discussed in detail, here only the determination of the correction factor is considered. The relevant plots are shown in fig. 5.28.
The correction factor is estimated to be $\sigma_\text{fit} = 1.36$, which is equivalent to a correlation of $\rho = 0.637$. The correlation is much higher than in the first iteration, which is due to the larger density of points which constrains the training more. It is also visible in the figure, that the observed error distribution deviates from the expected behavior, which indicates that the assumption of constant correlation is not strictly fulfilled. However, the uncertainty estimate is still reasonable, and because the uncertainty of the ensemble has only a minor impact on the posterior distribution it will not be investigated further.
To visualize and quantify the improvement the ensemble of the second iteration exhibits in comparison to the first iteration ensemble, their predictions can be plotted against the true values of the simulation for the test set. This is depicted in fig. 5.29 for three output p_T-bins. For an ideal emulator, the ensemble predictions would match the simulation outputs, resulting in all points laying on the diagonal.
For all three plots, the ensemble of the second iteration is closer to the ideal emulator model than the ensemble of the first iteration, which means that the prediction is more

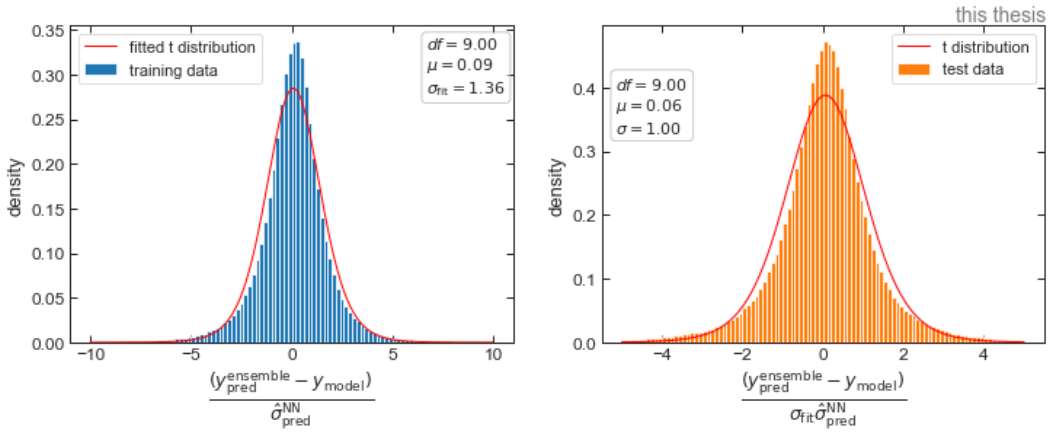


FIGURE 5.28: Left: Determination of the correction factor of the ensemble uncertainty by fitting a t-distribution. Right: Corrected distribution for the test data set. The distributions were obtained with the ensemble of the second iteration.

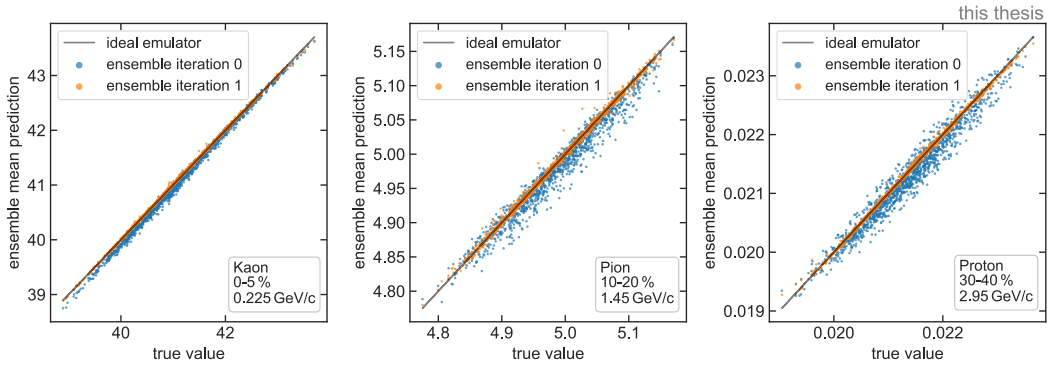


FIGURE 5.29: Comparison of the predictions of the ensemble from the first and second iteration. It is clearly visible, that the ensemble from the second iteration is much more accurate in its predictions.

accurate. In the left plot also a systematic underestimation of the first ensemble is visible, for the second iteration ensemble this is not the case. The observed improvement in the mean prediction is also apparent in all other 497 p_T-bins, which is not shown here.
The improvement also manifests in the corrected estimated uncertainties of the model. In fig. 5.30, the uncertainties of all p_T-bins of the test data samples of the ensemble of the second iteration are compared to the experimental uncertainties as well as the uncertainties from the ensemble of the first iteration.
The ensemble uncertainties of the second iteration are approximately five times smaller compared to the first iteration ensemble and are in the order of 0.5 % of the experimental uncertainty.
In conclusion, the newly trained ensemble is much more accurate and may therefore provide better estimates of the parameters.
The last step to infer the posterior estimates is to run the MCMC simulation, which is done analogously to the last section. The resulting distributions are depicted in fig. 5.31.
The produced distributions mostly agree with the ones from the first iteration. However, the marginal distributions for the normalization *Norm* and the two viscosities η/s and $(\zeta/s)_{\max}$ are more Gaussian shaped for this second iteration. This can be attributed to the lower impact of the ensemble uncertainty on the posterior probability.

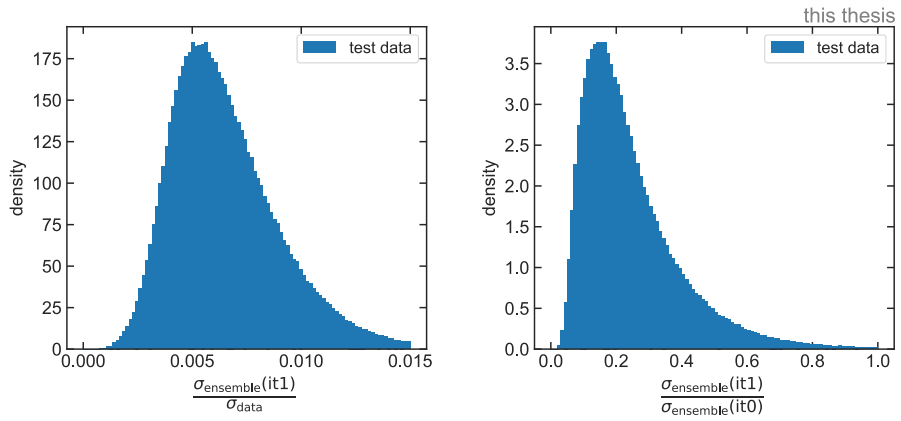


FIGURE 5.30: Left: Ensemble uncertainty compared to the experimental data uncertainty. Right: Ensemble uncertainty from the second iteration compared to the one from the first.

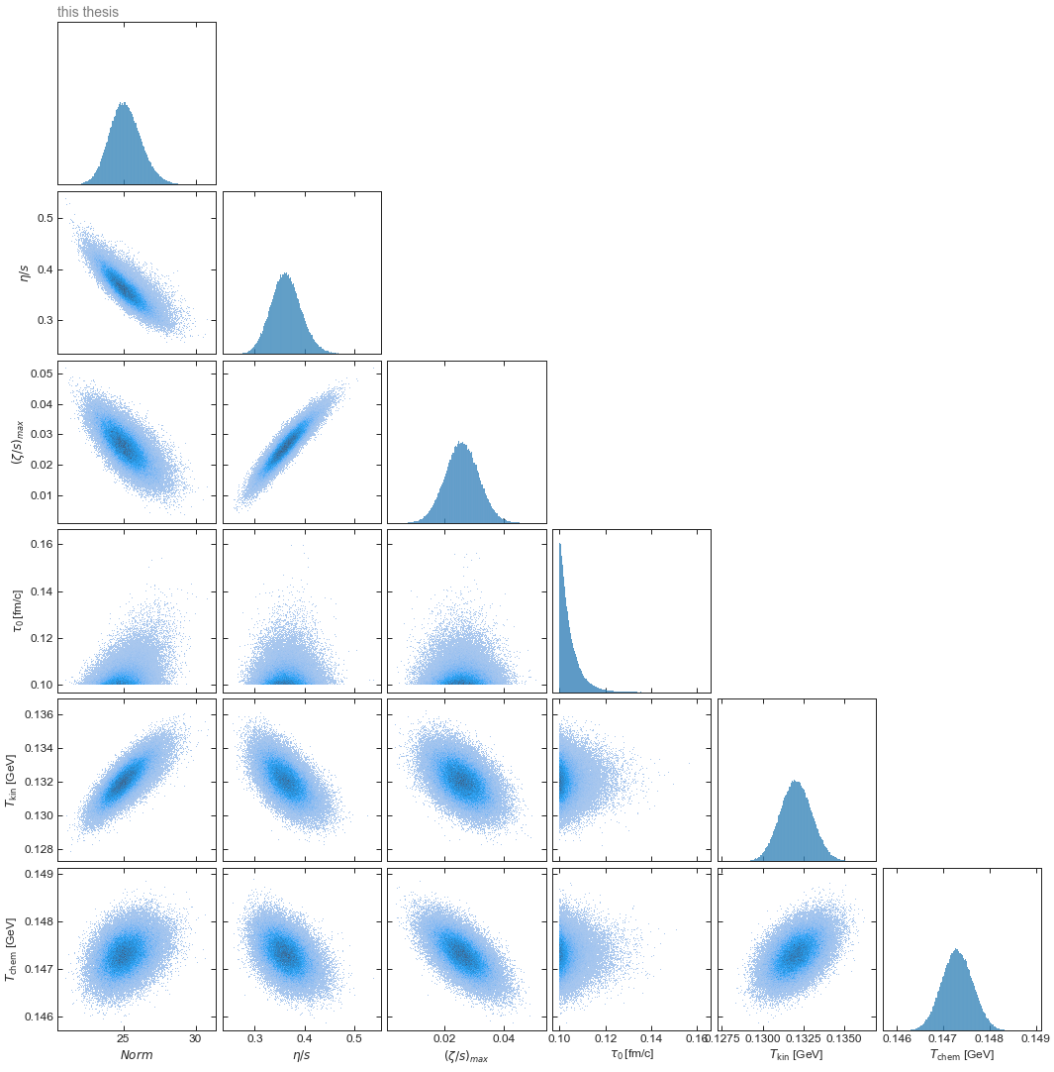


FIGURE 5.31: Marginal posterior probabilities for the input parameters inferred using the ensemble from the second iteration. Five considered centrality classes: 0-5 %, 5-10 %, 10-20 %, 20-30 %, and 30-40 %.

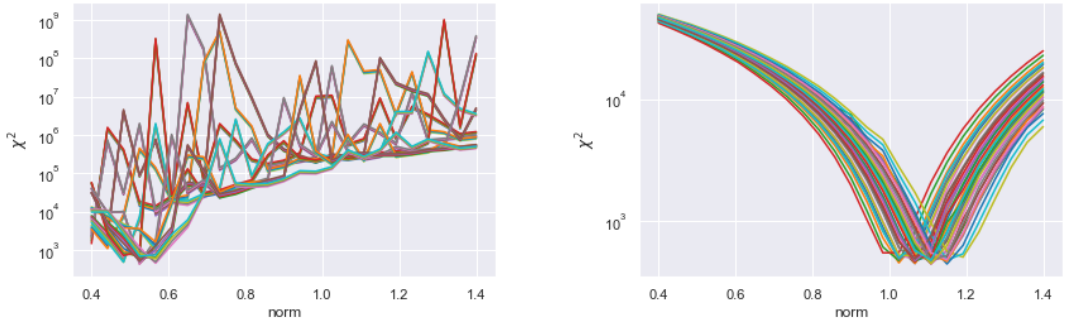
parameter	mean	median	68 % CI	95 % CI	99 % CI
$Norm$	25.1	25.1	24.1-26.2	23.1-27.4	22.5-28.2
η/s	0.36	0.36	0.33-0.39	0.31-0.42	0.29-0.45
$(\zeta/s)_{\mathrm{max}}$	0.026	0.026	0.020-0.032	0.015-0.037	0.012-0.041
τ_0 [fm/c]	0.104	0.103	0.101-0.109	0.100-0.118	0.100-0.125
T_{kin} [MeV]	132.1	132.0	131.1-133.0	130.2-134.0	129.6-134.6
T_{chem} [MeV]	147.3	147.3	147.0-147.6	146.7-148.0	146.5-148.1

TABLE 5.4: Mean, median, and credible intervals for posterior probability distributions of the parameter estimates. Fit based on five centrality classes (0-5 %, 5-10 %, 10-20 %, 20-30 %, and 30-40 %) and three particles (pions, kaons, protons).

To quantify the parameter estimates, credible intervals can be constructed using the marginal probability densities. A credible interval defines an interval within which a parameter value falls with a certain probability similar to confidence intervals in frequentist statistics. Here, the 68 %, 95 %, and 99 % credible intervals will be used, which means that the value of the respective parameter has a probability of 0.68, 0.95, or 0.99 to be in the interval. Credible intervals can be easily computed with central percentiles, so the 95 % credible interval is for example defined by the 2.5th and 97.5th percentiles. The credible intervals together with mean and median values of the marginal posterior probability distributions of the parameters are given in tab. 5.4.
The credible intervals are mostly symmetric around the mean estimated parameter value since they follow a normal distribution. Only for the thermalization time τ_0, this is not true as the distribution is shaped like an exponential. Because of the Gaussian-like distributions, the mean and median are also equivalent.
The values in tab. 5.4 mark the results of the parameter optimization for the considered five centrality classes and three particles. The values will be evaluated in detail in the discussion in the next chapter.

5.8 Optimization at thermalization times below 0.1 fm/c

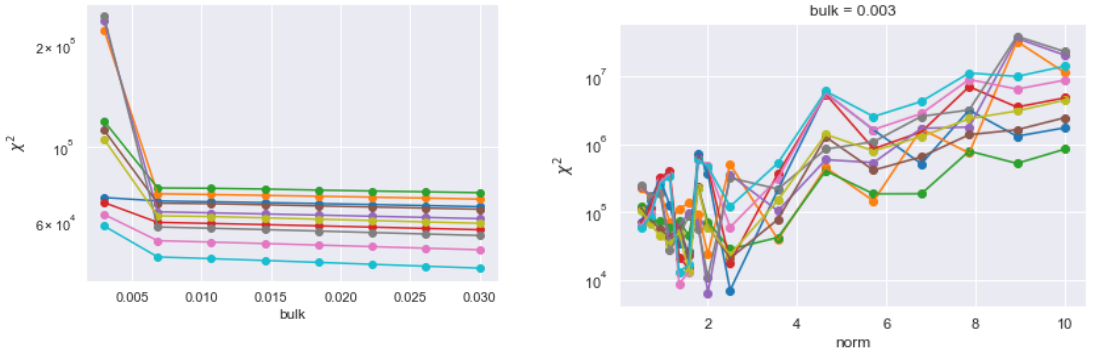
From the distributions shown in fig. 5.31, it is obvious that the optimal value of the thermalization time τ_0 is not included in the chosen search region, as the MCMC simulation runs into the lower boundary of $\tau_0 = 0.1\,\mathrm{fm/c}$. Before the physical implications are discussed, it is worthwhile to extend the search region to lower values, such that the optimum may be included. In principle, the extension of the grid to lower thermalization times is straightforward, as all it needs are new grid points in the area of $\tau_0 < 0.1\,\mathrm{fm/c}$ and then the introduced procedure may be employed. However, for the used simulation this is not possible because of one reason: For low thermalization times in the vicinity of zero, the simulation becomes unstable due to numerical instabilities. These distort the simulated output spectra of the particles and thus the posterior estimates heavily. The instabilities do not occur for all parameter configurations at low τ_0, but only for some specific ones. These regions are not predictable and hard to identify as there is no information on how the spectra should behave. One way to find these instabilities is by considering the χ^2-value of neighboring parameter configurations in the grid. Because the simulation output is continuous with respect to the input parameters, also the χ^2-value is continuous. For unstable regions, this is not the case, as can be seen as an example in the in figs. 5.32 and 5.33.



(A) Example of the unstable region for low τ_0. The χ^2-value is discontinuous.

(B) Example of the stable configurations for low τ_0. The χ^2-value changes continuously.

FIGURE 5.32: Normalization behavior.



(A) Example of instabilities for low values of the bulk viscosity $(\zeta/s)_{\max}$.

(B) Effect of the bulk instability on other parameters.

FIGURE 5.33: Bulk viscosity behavior.

Instabilities in the data also lead to a much worse fit of the ensemble model and become apparent in the correlation plots of the predictions (fig. 5.34). Therefore, checking for instabilities in the data can be done indirectly by checking these.

To infer the posterior estimates of the parameters for $\tau_0 < 0.1\,\mathrm{fm/c}$, the idea is now to iteratively expand the ranges of the search grid depending on where the MCMC simulation hits the boundary, train an ensemble, check if the region is stable and then infer again posterior estimates. This is done for the simulation with the same settings as before, such that only the parameter ranges are varied. Here only the final implications of this procedure will be outlined as the rest is similar to the last chapters.

From the procedure, two results were obtained: Firstly, the optimization for thermalization times below 0.1 fm/c led to larger posterior probabilities compared to those above it. This can be seen already from the minimum χ^2-values of the grid runs. For the analysis with the boundary at $\tau_0 = 0.1\,\mathrm{fm/c}$, the minimum χ^2-value was about 500, whereas the minimum χ^2-value for the grids at lower thermalization times was found to be 430 at a thermalization time of $\tau_0 = 10^{-5}\,\mathrm{fm/c}$. And secondly, the optimization always ran into the boundary of the lowest τ_0 for all generated grids, indicating that the optimal value for the thermalization time is zero, which is numerically infeasible to reach with the simulation. As an example, the posterior distribution for the lowest stable grid is shown in fig. 5.35, which has a boundary value of $\tau_0 = 10^{-4}\,\mathrm{fm/c}$.

To investigate the effect a smaller τ_0 has on the marginal posterior probabilities of the other parameters, the thermalization time was set to the smallest stable value of $\tau_0 = 10^{-4}\,\mathrm{fm/c}$ and the optimization procedure was employed by optimizing the other five

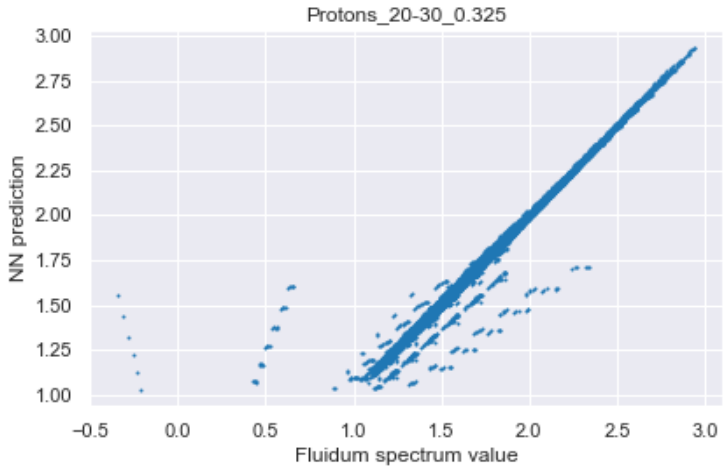


FIGURE 5.34: Effect of including numerically unstable regions in the NN fitting procedure. The unstable configurations are clearly fitted much worse than the stable points.

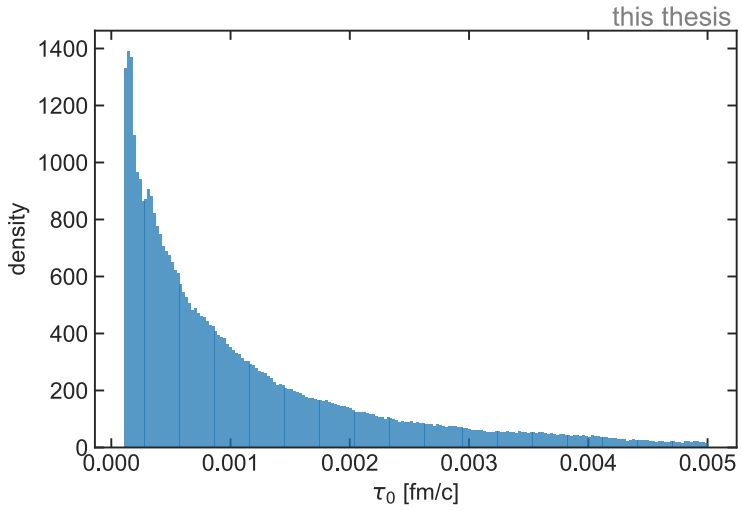


FIGURE 5.35: Marginal posterior probability distribution of the thermalization time at low τ_0. The optimal value is still not reached.

input parameters. The results are depicted in fig. 5.36.
The result is seemingly a multivariate normal distribution. In comparison to fig. 5.31, the distributions of $Norm$, η/s and $(\zeta/s)_{\text{max}}$ have shifted to lower values. In contrast to this, the distributions for the kinetic and chemical freeze-out temperatures remained mostly the same. Furthermore, the correlations between the parameters are similar to the ones observed in fig. 5.31 with the exception of correlations with the chemical freeze-out temperature, which are smaller here. To compare the differences quantitatively, the credible intervals are given in tab. 5.5.
The difference of the norm in tab. 5.4 and 5.5 is 23.7 and has an uncertainty of 1.0, thus the two values are significantly different. The differences for the shear and bulk viscosities are 0.07 ± 0.04 and 0.017 ± 0.006, so they are not significantly different but a shift can be notified. In contrast to this, the values for the kinetic freeze-out temperature T_{kin} and for the chemical freeze-out temperature T_{chem} fully agree ($\Delta T_{\text{kin}} = 0.1 \pm 1.4$, $\Delta T_{\text{chem}} = 0.3 \pm 0.4$). The implications of the findings in this section will be discussed in the next chapter.

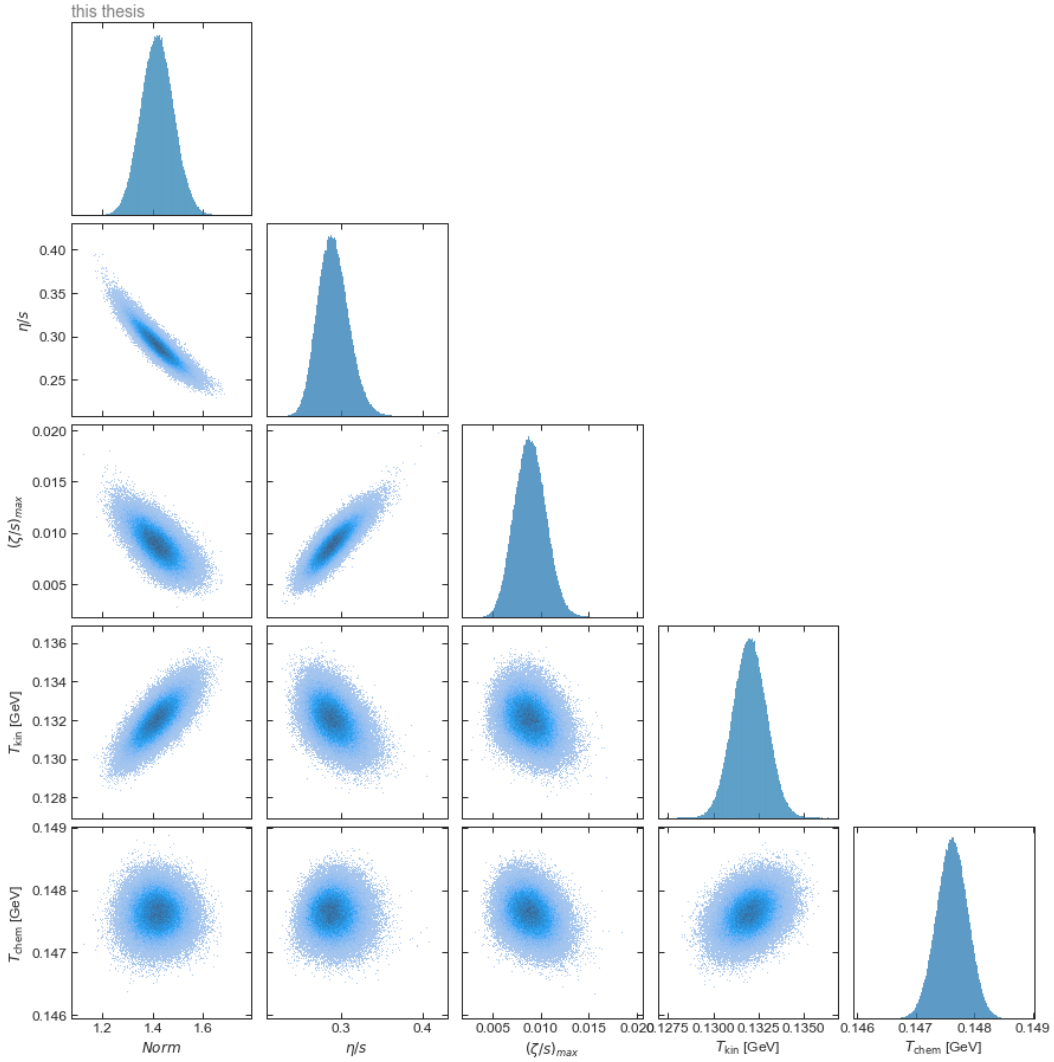


FIGURE 5.36: Marginal posterior probability distributions for $Norm$, η/s, $(\zeta/s)_{\mathrm{max}}$, T_{kin} and T_{chem} for a fixed $\tau_0 = 10^{-4}$ fm/c for the centrality classes 0-5 %, 5-10 %, 10-20 %, 20-30 %, and 30-40 % for pions, kaons, and protons.

parameter	mean	median	68 % CI	95 % CI	99 % CI
$Norm$	1.42	1.42	1.36-1.48	1.30-1.55	1.26-1.59
η/s	0.29	0.29	0.27-0.31	0.26-0.33	0.25-0.34
$(\zeta/s)_{\mathrm{max}}$	0.009	0.009	0.007-0.011	0.006-0.012	0.005-0.013
T_{kin} [MeV]	132.0	132.0	131.1-133.0	130.2-133.8	129.7-134.4
T_{chem} [MeV]	147.6	147.6	147.4-147.9	147.1-148.1	146.9-148.3

TABLE 5.5: Mean, median, and credible intervals for the posterior probability distributions of the parameter estimates for a fixed $\tau_0 = 10^{-4}$ fm/c. Fit based on five centrality classes (0-5 %, 5-10 %, 10-20 %, 20-30 %, and 30-40 %) and three particles (pions, kaons, protons).

5.9 An alternative ansatz for the emulator

During the development of the optimization procedure laid out in the last chapters, an alternative ansatz for the emulator was also tested, which has not been mentioned yet. For completeness, it will be discussed here.
As already explained in 5.6, the likelihood function, and therefore also the posterior probability, is effectively dependent on the χ^2-value between the simulation output and the experimental data. In the previous analysis, this value was calculated for a specific parameter configuration with the emulated simulation output and the experimental data after the emulation. However, since the experimental data is always the same, it is possible to include the calculation of the χ^2 into the emulator. This means, that the emulator may not be used to predict the output spectra of the considered particles, but directly the χ^2-value for a specific parameter configuration. This methodical shift is illustrated in fig. 5.37.

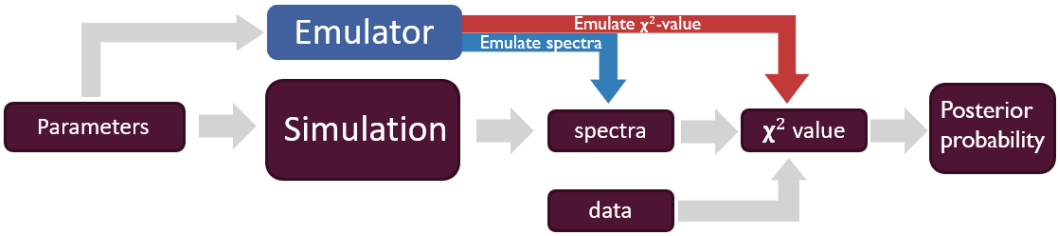


FIGURE 5.37: Instead of emulating the particle spectra and subsequently deriving the χ^2-value, both steps can be included in the emulator model.

The change in the emulator model does not make any conceptual difference, hence all derived posterior probability estimates stay the same. Nevertheless, emulating the χ^2-value exhibits some advantages and disadvantages. An advantage of it is that the uncertainty of the posterior probability is expected to be lower. This is because while emulating the spectra, the model uncertainty of all 500 p_T-bins add up, whereas for the χ^2-emulator only one output node is considered. A disadvantage of the approach is, that the information of the 500 p_T-bins is condensed to only one value, which makes the output space much more complex and thus the training of the emulator.
To prove that emulating the χ^2 results in the same posterior parameter estimates as for the spectra emulator, the introduced optimization procedure is adapted and deployed for its use. The analysis is started from the second iteration grid generated with the simulation. Because of this, it is not fully independent of the first analysis with the spectra emulation model, as only the region of the highest probability of the first iteration is considered. This is done because the first grid has a density of points that is too low to fit the χ^2-emulator accurately. The effect occurs since the χ^2 that is fitted has a more complex dependence on the input parameters than the individual 500 spectra values. Nevertheless, if the estimates differ from the previous results, this will be visible in the posterior distributions. In the following, the adaptions of the procedure for using the χ^2-emulator are discussed.

5.9.1 Parameter grid

The parameter grid is the same as used in the previous analysis, but here directly the second iteration grid is used. All outputs for the particle spectra of a specific input parameter configuration are condensed into one χ^2-value using the experimental data. The χ^2-values are taken as the new outputs. The division into train, validation, and test set remains the same.

5.9.2 Emulator model

For the emulator, the data preprocessing also remains the same. However, the architecture and hyperparameters of the NN change. The optimal settings of these were inferred by trial and error, without grid search. They can be found in tab. 5.6.

hyperparameter	value	hyperparameter	value
input nodes	6	output nodes	1
# layers	8	# nodes per layer	16
activation function	Tanh	weight initialization	Xavier
Number of epochs	300	Batch size	100
initial learning rate	0.001	loss	MSE

TABLE 5.6: Hyperparameters of the NN for the χ^2-fit.

The found hyperparameters are quite different from the ones for the spectra fit. Instead of a wide shallow NN, a deeper NN with 8 layers worked better here, which may be related to the more complex structure of the output space. Furthermore, the hyperbolic tangent performed better than the ReLU activation function. Additionally, the predictive performance of the NN was improved by using a learning rate scheduler, which reduces the learning rate if the loss reaches a plateau, and oversampling of the training set, such that in each epoch the training set is seen multiple times. The training and validation loss of the used NN are shown in fig. 5.38.

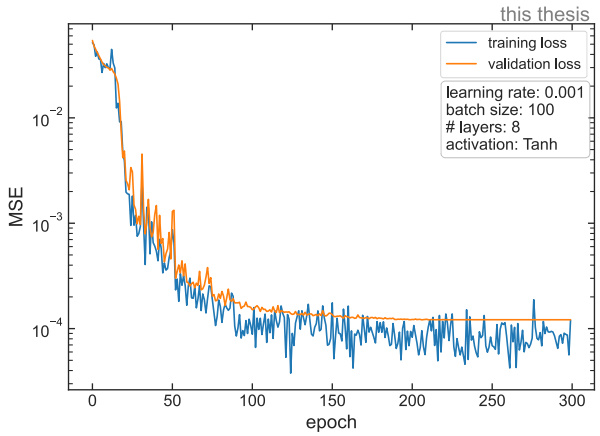


FIGURE 5.38: The training and validation loss of the constructed NN fitting the χ^2 of the simulation to data.

With the NN in place, the ensemble can be constructed by combining the predictions of several NNs. Since the principles remain the same, this is done analogously to the case of the spectra fit. The ensemble consists of 10 NNs and its uncertainty can be estimated again by the spread of the predictions. The resulting correlation plot for the constructed χ^2-ensemble is depicted in fig. 5.39.
From the left plot of the figure, it is evident that the χ^2-values predicted by the ensemble reproduce the true χ^2 from the simulation. Moreover, from the right plot, it es apparent that the uncertainty of the ensemble is distributed around zero with a difference in the predicted and the true χ^2 of mostly below 1.
Using the ensemble, the correction factor can be estimated again by fitting a Student's t-distribution to the error distribution. The corresponding plots are shown in fig. 5.40. The estimated correction factor is $\sigma_{\text{fit}} = 0.49$, which is equivalent to a mean correlation between the networks of $\rho = 0.12$. This is much smaller than for the spectra ensemble

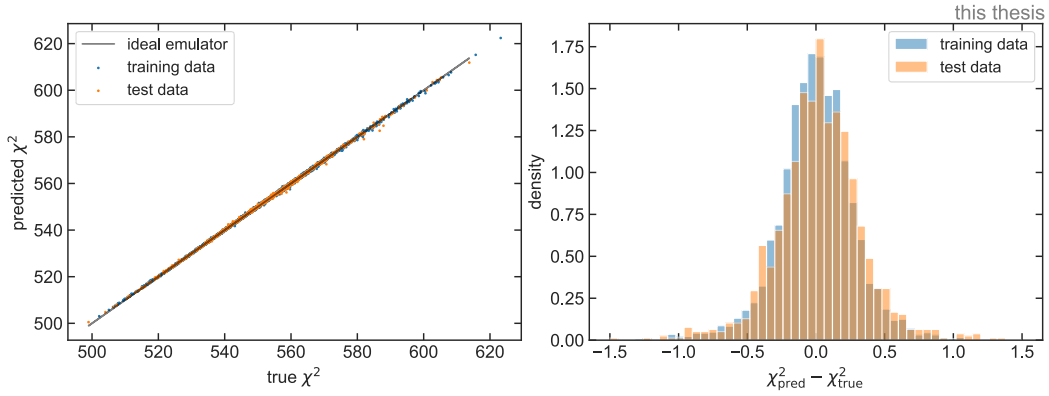


FIGURE 5.39: Left: Correlation between χ^2-value calculated with the simulation and predicted with ensemble model. Right: Error distribution for the χ^2-value.

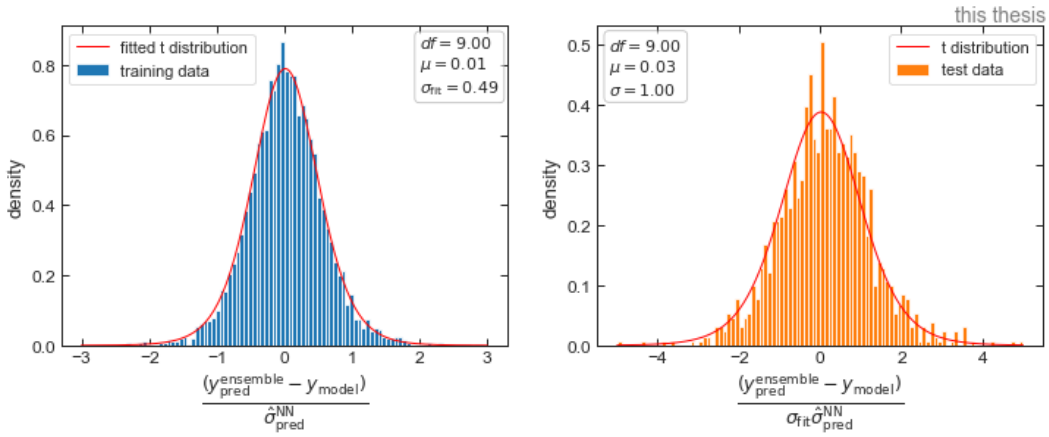


FIGURE 5.40: Left: Fit of a t-distribution to the error distribution of the training data. Right: The corrected test data and the expected t-distribution.

($\rho = 0.64$) trained on the same data, which may be attributed to the changed network architecture and the strong correlation between the spectra outputs.

5.9.3 MCMC simulation

Having constructed and verified the ensemble model, the MCMC simulation can be run. However, at first, some important changes in the expression of the posterior probability have to be discussed. For the spectra fit, it is given by eq. 5.27, where the uncertainties of the experiment and the model are both captured in the covariance matrix Σ. This is not the case for the χ^2-ensemble, as the χ^2, given by

$$\chi^2 = [\mathbf{y}_m(\mathbf{x}) - \mathbf{y}_e]^{\mathrm{T}} \Sigma^{-1}_{\text{exp}} [\mathbf{y}_m(\mathbf{x}) - \mathbf{y}_e] , \tag{5.21}$$

only captures the experimental uncertainty and the ensemble uncertainty comes in at a later stage as the prediction uncertainty of this value. For the ensemble model, the logarithmic posterior probability is given by

$$\log(P(D|\mathbf{x})) \propto \begin{cases} -\frac{1}{2}\chi^2 & \text{if } x_i^{\min} \leq x_i \leq x_i^{\max} \,\forall\, i \\ -\infty & \text{else} \end{cases} . \tag{5.22}$$

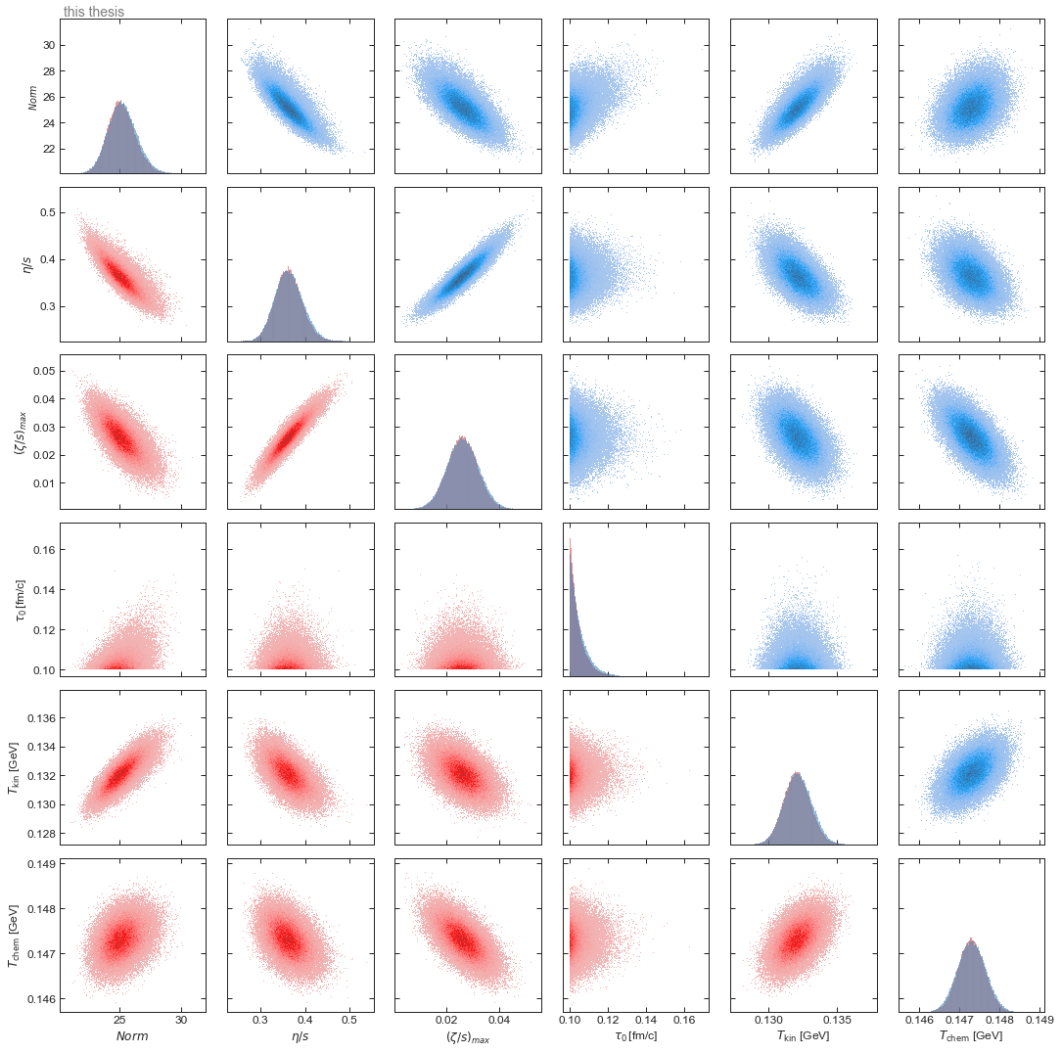


FIGURE 5.41: The marginal and joint posterior distributions for all considered input parameters. Blue: χ^2-ensemble, red: spectra-ensemble.

Because the χ^2 is not exactly known but has to be predicted, it has to be replaced by its estimate, which is given by a normal distribution around the ensemble mean prediction χ^2_{pred} with a standard deviation of $\hat{\sigma}_{\text{pred}}$. Thus, the posterior probability for a fixed set of parameters $\mathbf{x}$ follows a log-normal distribution. The MCMC simulation is not able to process distributions, but only scalar values. Hence, the expectation value for the posterior probability is used here, which is given by the log of the expectation value of the log-normal distribution $\chi^2_{\text{pred}} + 1/2\hat{\sigma}^2_{\text{pred}}$. Using this, the expression for the log of the expectation of the posterior probability becomes

$$\log \mathbb{E}(P(D|\mathbf{x})) \propto \begin{cases} -\frac{1}{2}\chi^2_{\text{pred}} + \frac{1}{8}\hat{\sigma}_{\text{pred}} & \text{if } x_i^{\min} \leq x_i \leq x_i^{\max} \,\forall\, i \\ -\infty & \text{else} \end{cases}. \tag{5.23}$$

This is used for the MCMC simulation. The simulation is run with the same settings as before (see sec. 5.6). The results of the MCMC simulation are shown in fig. 5.41.
For comparison, in fig. 5.41, in addition to the marginal distributions of the χ^2-ensemble in blue, the results of the spectra-ensemble are depicted in red. It is evident, that both methods converge to the same solution. This is the expected behavior, as both variants

calculate the same quantity, the posterior probability, and are based on the same experimental data and simulation. Differences may only arise from the different treatments of the errors or uncertainties in the prediction. Because major disagreements do not occur, it can be concluded, that the model uncertainties are small enough to have no effect on the posterior distributions, and it indicates that the estimates can not be further refined by improving the emulator model. The findings will be discussed in more depth in the next chapter.

5.10 Parameter optimization for the most central events

The performed optimization of the last chapter led to a vanishing thermalization time $\tau_0 < 10^{-4}\,\mathrm{fm/c}$, which is incompatible with the physical expectation. This indicates, that the simulation misses a part of the description of the evolution of the heavy-ion collision or introduces an error that is not considered. In correspondence with the theory group which is responsible for the simulation of Trento, Fluidum, and FastReso, two possible reasons for the small thermalization time could be identified: Firstly, within the model, the Equation of State is applied only to the averaged background fields. By that, some contribution to the energy of the fluctuations of the individual collisions is lost, which introduces a source of error. The effect is expected to be small in the most central collisions as for them the fluctuations are small. However, for more peripheral collisions, the approximation is expected to be less reliable which may drive τ_0 to zero. The second reason could be that the fluid velocity is set to zero as an initial condition for the fluid dynamic evolution. Thus, to build up the fluid velocity that is needed to reproduce the data, the optimization is driven to particularly low thermalization times. The issues may be solved by modeling the pre-equilibrium phase of the heavy-ion collision explicitly, such that a fluid velocity can build up and more realistic initial conditions arise. This will be implemented in the near future into the simulation, but was not yet available for this work. To nevertheless resolve the first presented issue, the optimization is done considering only the most central events, where the error connected to the linearization is small.
Optimizing only considering the spectra of the first centrality class from $0-5\,\%$ is straightforward. The whole procedure for parameter optimization that was introduced in this thesis stays the same, but now only the 100 p_T-bins of the first centrality class of the three particles are considered. It is even possible to reuse the first generated parameter grid, as all information about the outputs of the simulation of the $0-5\,\%$ centrality class is included in it. Moreover, since the fit to the first centrality class is a part of the previous optimization, all assumptions about its uncertainty remain valid. Therefore, the emulator model is trained on the outputs of the first parameter grid with the same hyperparameters as for five centrality classes. The correction factor of the uncertainty can be inferred again by fitting a t-distribution. The corresponding plots are depicted in fig. 5.42.
The estimated correction factor is $\sigma_\mathrm{fit} = 0.48$, which is equivalent to a mean correlation of $\rho = 0.115$. The correlation is, as expected, in the same magnitude as for the first ensemble ($\rho = 0.081$), because the fit is performed on a fraction of the same data. Having defined the ensemble model, the MCMC simulation can be performed analogously to the previous runs. For considering only the first $0-5\,\%$ centrality class, the obtained results for the first iteration are given in fig. 5.43.
The thermalization time τ_0 seems to be more constrained for the new settings. However, before the distributions are evaluated in more detail, a second iteration of the optimization is performed to improve the results. This is done similarly to the procedure in 5.7. The threshold was set to $p_\mathrm{threshold} = -44$, which was estimated based on

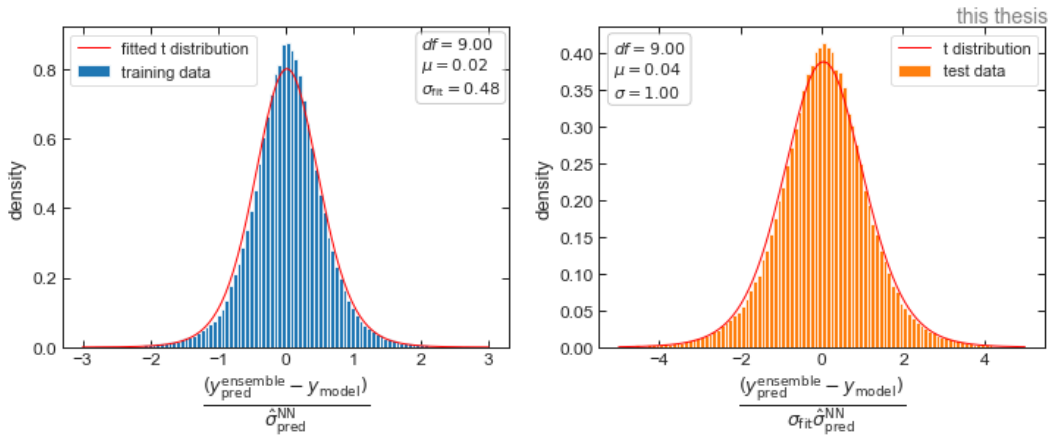


FIGURE 5.42: Left: Error distribution for training data of ensemble fitting only the $0-5\,\%$ centrality class. Right: Corrected distribution for the test data.

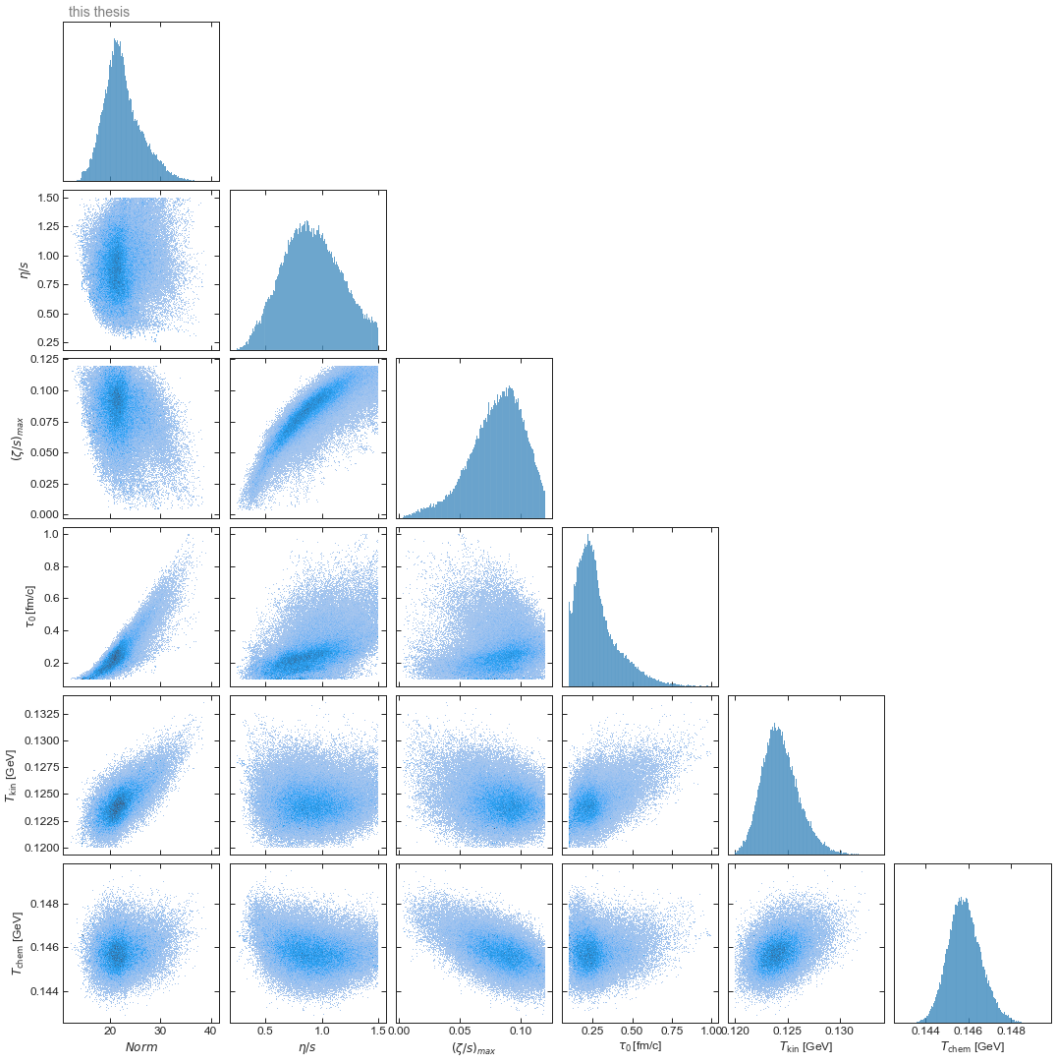


FIGURE 5.43: Marginal posterior probability distributions and correlations for the optimization for the $0-5\,\%$ centrality class for pions, kaons, and protons.

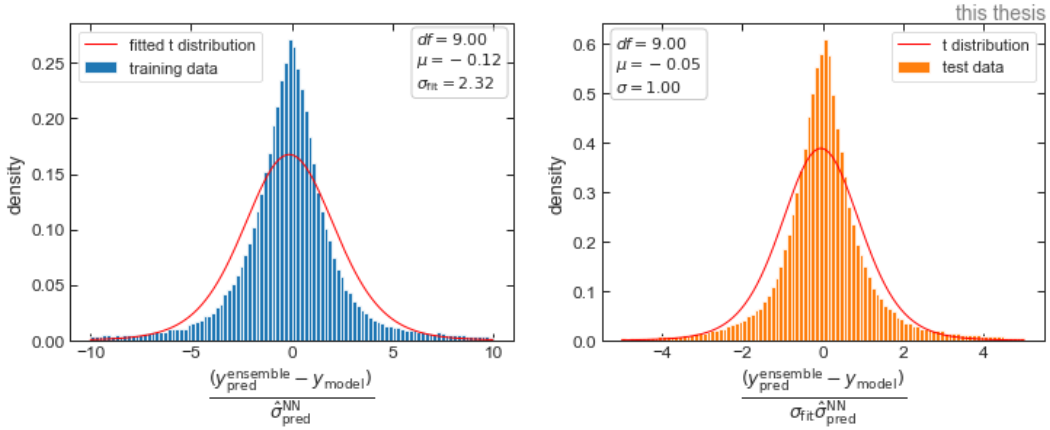


FIGURE 5.44: Left: Fit of the t-distribution to the error distribution of the second iteration ensemble. Right: The corrected error distribution of the test data.

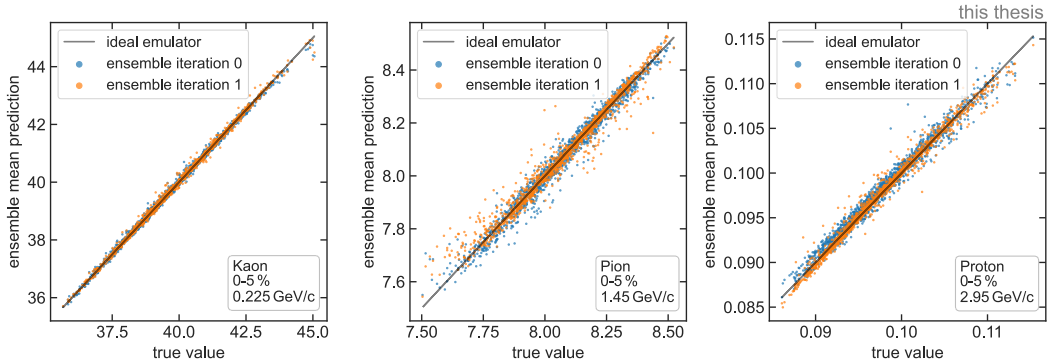


FIGURE 5.45: The correlation between the ensemble model prediction and the simulation output for three different p_{T}-bins. The ensemble from the second iteration is only slightly better than the one from the first iteration.

the distribution of the logarithmic posterior probability from the first iteration. With the procedure, 7360 new grid points were sampled in the region of the highest probability. These were then used to train and test a new ensemble model. For this, the correction factor can be determined again as illustrated in fig. 5.44.
The correction factor is estimated to be $\sigma_{\mathrm{fit}} = 2.32$, which is connected to a mean correlation of $\rho = 0.84$. From the figure, it is obvious that the error distribution is not described well by a t-distribution, which indicates that the correlation is not constant everywhere. However, the corrected uncertainty may still be used as an estimate for the uncertainty as it defines an upper boundary of the real uncertainty. The mean correlation between the NNs is rather large, even larger than for the case of five centrality classes ($\rho = 0.64$), which means that combining the predictions of several networks improves their prediction only marginally. This is also visible in the comparison of the ensemble prediction and the true simulation output depicted in fig. 5.45.
For the three shown p_{T}-bins, the second iteration ensemble is only slightly better than the first iteration ensemble. For pions at 1.45 GeV/c there are even more outliers for the second iteration ensemble than for the first one. The marginal improvement may be attributed to the strong correlation and to the lower density of points in the region of the highest probability. The density of points is lower than for the case of the five centrality classes because a similar number of samples ($\sim$ 5000) is distributed in a larger space.

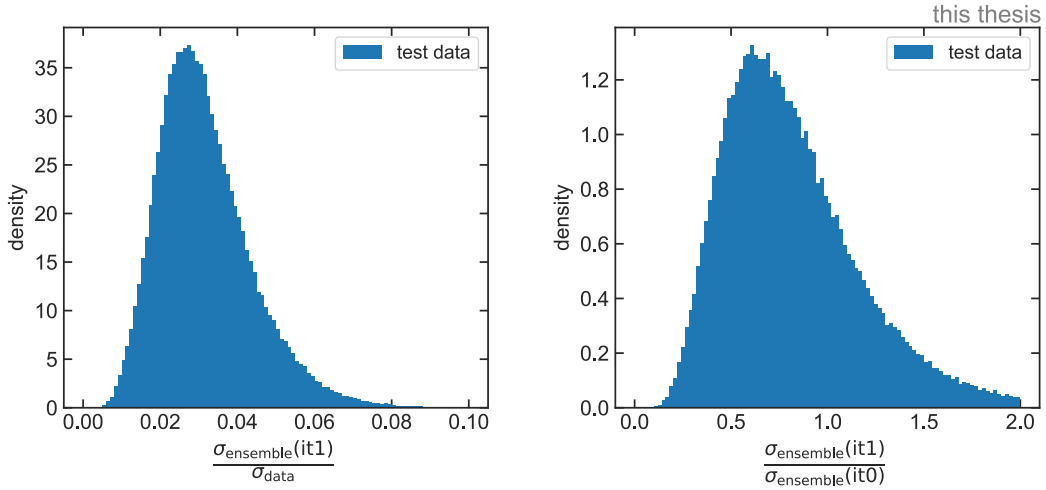


FIGURE 5.46: Uncertainties of the second iteration ensemble compared to Left: the experimental uncertainties, Right: the uncertainties of the first iteration ensemble.

For example, the ranges for the viscosities obtained from the first iteration cover their whole search regions instead of only a small part as for the case of the five centrality optimization.
To evaluate the uncertainties, they can be compared again to the experimental ones and the ones from the first iteration. This is depicted in fig. 5.46.
The estimated and corrected uncertainties are in the order of 3 % of the experimental uncertainties. This is about six times larger than in the five-centrality fit. Furthermore, the distribution in the left plot confirms the results of fig. 5.45. On average, the second iteration ensemble is slightly better than the first one, but for some points, it is worse with uncertainties up to twice as large as in the first iteration. The ensemble will be used nonetheless to infer again the posterior probability densities. This is done analogously to sec. 5.6. The resulting distributions are given in fig. 5.47.
The distributions for the kinetic and the chemical freeze-out temperature are approximately normally distributed, whereas all other distributions follow more complex shapes. The viscosities η/s and $(\zeta/s)_{\text{max}}$ are distributed over the whole search range and tend, in contrast to the fit to five centralities, to the larger values. Both distributions are cut off at the boundaries, which may also affect the other parameter estimates. The distribution of the thermalization time runs again to the lower boundary at $\tau_0 = 0.1\,\text{fm/c}$, which indicates the problem of the vanishing thermalization time is more related to the fluid velocity of zero in the initial conditions. However, the increase to the boundary is less pronounced than for the fit to five centrality classes, which suggests that the error induced by linearization has an effect on τ_0. The mean estimates and credible intervals for the parameters are given in tab. 5.7.
The derived credible intervals and means of the distributions have to be taken with care because the posterior probability is cut off in some dimensions. If the search grid is extended, the estimates could potentially change. The obtained parameter estimates are only true if the underlying assumption of the analysis holds, that the optimal parameters are within the search ranges. The results are discussed now in more detail in the discussion.

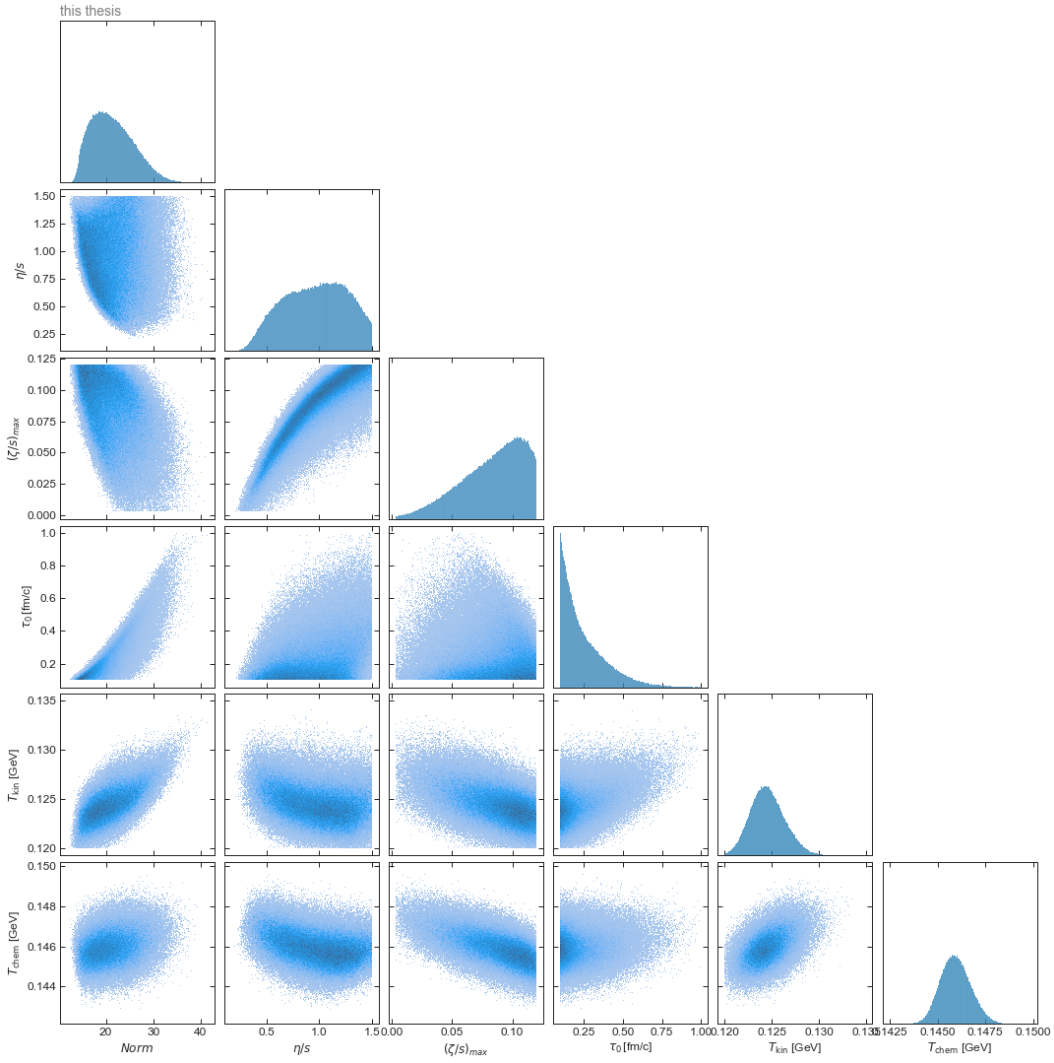


FIGURE 5.47: Marginal posterior probability distributions for the optimization based on the spectra of pions, kaons, and protons for the centrality class $0-5\,\%$.

parameter	mean	median	68 % CI	95 % CI	99 % CI
$Norm$	21.4	20.9	16.8-26.0	14.6-30.8	13.8-33.5
η/s	0.96	0.97	0.63-1.28	0.41-1.45	0.33-1.49
$(\zeta/s)_{\mathrm{max}}$	0.08	0.09	0.05-0.11	0.02-0.12	0.01-0.12
τ_0 [fm/c]	0.25	0.21	0.13-0.39	0.10-0.61	0.10-0.76
T_{kin} [MeV]	124.5	124.4	122.7-126.4	121.3-128.4	120.5-129.8
T_{chem} [MeV]	146.0	146.0	145.2-146.7	144.5147.6	144.1-148.1

TABLE 5.7: Mean, median, and credible intervals for posterior probability distributions of the parameter estimates. The fit is based on one centrality class (0-5 %) and three particles (pions, kaons, protons).

Chapter 6

Results and discussion

The final results of the parameter estimates inferred in this work for the normalization *Norm*, the shear viscosity η/s, the maximum of the bulk viscosity $(\zeta/s)_{\mathrm{max}}$, the thermalization time τ_0, the kinetic freeze-out temperature T_{kin}, and the chemical freeze-out temperature T_{chem} are summarized in tab. 6.1. The table includes the optimized parameters for the three considered cases: Firstly, the fit to the spectra of pions, kaons, and protons in five centrality classes (0-5 %, 5-10 %, 10-20 %, 20-30 %, and 30-40 %) with a limiting τ_0 of 0.1 fm/c. Secondly, the fit to the spectra in five centrality classes (0-5 %, 5-10 %, 10-20 %, 20-30 %, and 30-40 %) with a fixed $\tau_0 = 10^{-4}$ fm/c. And lastly, the fit to the spectra in the centrality class $0-5$ % with a limiting τ_0 of 0.1 fm/c. For all three cases, pions were fitted in the pp-range $0.5-3.0$ GeV, and kaons and protons for $p_{\mathrm{T}} < 3.0$ GeV.

parameter	mean	median	68 % CI	95 % CI	99 % CI
pions, kaons, protons, centrality classes: 0-5 %, 5-10 %, 10-20 %, 20-30 %, 30-40 %					
Norm	25.1	25.1	24.1-26.2	23.1-27.4	22.5-28.2
η/s	0.36	0.36	0.33-0.39	0.31-0.42	0.29-0.45
$(\zeta/s)_{\mathrm{max}}$	0.026	0.026	0.020-0.032	0.015-0.037	0.012-0.041
τ_0 [fm/c]	0.104	0.103	0.101-0.109	0.100-0.118	0.100-0.125
T_{kin} [MeV]	132.1	132.0	131.1-133.0	130.2-134.0	129.6-134.6
T_{chem} [MeV]	147.3	147.3	147.0-147.6	146.7-148.0	146.5-148.1
pions, kaons, protons, five centrality classes, fixed $\tau_0 = 10^{-4}$ fm/c					
Norm	1.42	1.42	1.36-1.48	1.30-1.55	1.26-1.59
η/s	0.29	0.29	0.27-0.31	0.26-0.33	0.25-0.34
$(\zeta/s)_{\mathrm{max}}$	0.009	0.009	0.007-0.011	0.006-0.012	0.005-0.013
T_{kin} [MeV]	132.0	132.0	131.1-133.0	130.2-133.8	129.7-134.4
T_{chem} [MeV]	147.6	147.6	147.4-147.9	147.1-148.1	146.9-148.3
pions, kaons, protons, centrality class 0-5 %					
Norm	21.4	20.9	16.8-26.0	14.6-30.8	13.8-33.5
η/s	0.96	0.97	0.63-1.28	0.41-1.45	0.33-1.49
$(\zeta/s)_{\mathrm{max}}$	0.08	0.09	0.05-0.11	0.02-0.12	0.01-0.12
τ_0 [fm/c]	0.25	0.21	0.13-0.39	0.10-0.61	0.10-0.76
T_{kin} [MeV]	124.5	124.4	122.7-126.4	121.3-128.4	120.5-129.8
T_{chem} [MeV]	146.0	146.0	145.2-146.7	144.5147.6	144.1-148.1

TABLE 6.1: Mean, median, and credible intervals for posterior probability distributions of the parameter estimates. Summary of the three analyses.

6.1 Optimization for five centrality classes

For the optimization based on the spectra of pions, kaons, and protons in five centrality classes, the posterior distributions are shown in fig. 5.31. Before the derived estimates are discussed in detail, some important properties of them have to be kept in mind: The derived posterior distributions are only a valid description of the global optimal parameters if the assumption that was made by defining the prior, that the probability of the true value laying outside the search range is zero, is true. This is not fulfilled here, because the thermalization time converges to values at the defined limit of its range. Therefore the obtained parameter values may significantly change with respect to the true global optimum. However, as the search range was expanded to lower thermalization times, the posterior distributions could also be studied near the potential optimum at $\tau_0 = 0\,\mathrm{fm/c}$, thus the effect of the thermalization on the other parameters can be estimated.
Another important aspect is, that the uncertainties of the parameter estimates in this thesis only consist of the experimental and the emulator uncertainties, but do not include systematic uncertainties of the simulation model, which are expected to contribute significantly to the overall error, similar to the results of [24]. These can be estimated by varying the fitting procedure and this will be done in future work. Having said this, the inferred parameter estimates can be compared to the results of other hydrodynamic modeling approaches, especially to the predecessor of this work [24]. Since the used simulation model has undergone some important changes with respect to [24], a detailed comparison is however not possible.
The optimal *Norm* parameter is 25.1 ± 1, which is significantly different from the analysis in [24], which found a normalization of about 56. The difference most certainly comes from the fact, that [24] only searched the optimal *Norm* parameter in the range 50-67, whereas in this work the range was extended also to lower values after prior studies have shown that this produces better fits. Extending the τ_0 range to lower values near zero revealed a strong correlation between the *Norm* and the thermalization time. For a fixed thermalization time $\tau_0 = 10^{-4}$ the *Norm* was reduced to a value of 1.42 ± 0.05, significantly different to the other value. The strong positive correlation occurs due to the scaling of the entropy profiles according to eq. 5.1 by a factor of $Norm/\tau_0$. Because of the strong dependence between the normalization and the thermalization time, the estimate for the *Norm* is considered unreliable until the problem of the vanishing τ_0 is fixed.
The value for the shear viscosity $\eta/s = 0.36 \pm 0.03$ is well above its postulated lower boundary at $1/4\pi$. Compared to other works, the determined value is rather large, [59] reported a value of $\eta/s = 0.095$, [24] a value of $\eta/s = 0.164^{+0.079}_{-0.07}$, and [23] a temperature-dependent value of the same order. The results of this thesis and [24] are not significantly different.
The found optimal value for the maximum bulk viscosity $(\zeta/s)_{\mathrm{max}} = 0.026 \pm 0.06$ is consistent with the results of [24] and [6], all ranging in a region of 0.01-0.06. The value of 0.3 from [59] could not be confirmed. For a lower thermalization time at $\tau_0 = 10^{-4}\,\mathrm{fm/c}$, the shear and bulk viscosities decrease to $\eta/s = 0.29 \pm 0.02$ and $(\zeta/s)_{\mathrm{max}} = 0.009 \pm 0.002$. Similar to the findings of [24], the two viscosities exhibit a strong positive correlation.
For the thermalization time τ_0, the posterior probability distribution was found to converge to zero, which is incompatible with other estimates and the physical expectation. As already explained in sec. 5.10, this is expected to be related to two issues: A linearization error arising from not including the contribution of fluctuations for the energy, and the zero fluid velocity as an initial condition of the fluid dynamic evolution.

The first point was addressed in this thesis by fitting only the most central events where the error is expected to be small and is discussed in the next section. To solve the second issue, in the future, a model for the pre-equilibrium phase of the heavy-ion collision is included in the simulation model. The optimization may then provide more reliable results for the thermalization time.
The kinetic freeze-out temperature is found to be $T_{\mathrm{kin}} = 132.1^{+0.9}_{-1.0}\,\mathrm{MeV}$ and the chemical one to be $T_{\mathrm{chem}} = 147.3 \pm 0.3\,\mathrm{MeV}$, which agrees with the result for thermal fits to particle yields in [60]. In [24], a freeze-out temperature of $T_{\mathrm{fo}} = 137.1^{+8.0}_{-2.8}\,\mathrm{MeV}$ was reported, which is right between both temperatures and is consistent with both. The temperatures stay the same for the reduced value of the thermalization time of $\tau_0 = 10^{-4}\,\mathrm{fm/c}$, they seem to be independent of it. This strengthens the expressive power of the estimates, as the issue of the thermalization time seems to not affect them.
In fig. 6.1, the optimal solution of the particle spectra obtained with the simulation model is compared to the experimental data. In the upper panels, the Maximum-a-posteriori (MAP) estimate is shown as the optimal solution. This is the mode of the posterior probability distribution or in other words, the most probable set of parameters. The lower panels depict the data-to-model ratios, where the uncertainty bands were constructed by evaluating the simulation model for samples drawn from the posterior probability distribution of the parameters. The simulations mostly agree with the data within an accuracy of $10 - 20\,\%$, except for high-p_{T} pions, which exhibit larger ratios of up to 1.8. The fit is consistent with the experimental data for protons within the error margins, for pions and kaons there are significant deviations. The shape of the ratios is similar to [24]. For pions, there is a tension visible between the data and the model, which increases for larger centralities. Pions at $p_{\mathrm{T}} > 2.5\,\mathrm{GeV/c}$ in peripheral events are underestimated by the model, which indicates that the simulation model does not describe the production of pions completely.

6.2 Optimization based on the most central events

In the second part of the analysis, the optimization was performed based on the particle spectra of pions, kaons, and protons in the first centrality class from $0 - 5\,\%$. The resulting posterior distributions are depicted in fig. 5.47. For the derived estimates, the same remarks apply as in the last section. In this case, the posterior probabilities are less reliable, because in addition to the thermalization time the distributions for the shear and bulk viscosity are not fully covered. This may also alter the estimates for the other parameters. Therefore, the quantitative estimates have to be taken with care. At least for the shear and bulk viscosities, the issue can be fixed by performing the optimization in a larger search space, which was not done here due to time constraints.
The posterior distributions exhibit a more complex structure than for the fit to five centrality classes. For example, the joint probability distribution of the shear and bulk viscosities is bent or the marginal distribution for the normalization is skewed. This could be because the posterior captures a larger region of the parameter space, such that the more complex large-scale behavior of the simulation is captured. If the posterior would be localized to a small region, it could be approximated by simpler distributions similar to the local approximation of complex functions by easier ones.
The value of the $Norm = 21.4^{+4.6}_{-4.6}$ is consistent with the value of the last section. The shear viscosity $\eta/s = 0.96^{+0.32}_{-0.33}$ is very large, however, because its uncertainty is also large, it agrees with the estimate for the fit in five centrality classes. The same is true for the maximum bulk viscosity $(\zeta/s)_{\mathrm{max}} = 0.08 \pm 0.03$. Hence, excluding more peripheral centrality classes seemingly loosens the constraining power of the fit. The thermalization time is again converging to the lower limit of the search region of $\tau_0 = 0.1\,\mathrm{fm/c}$,

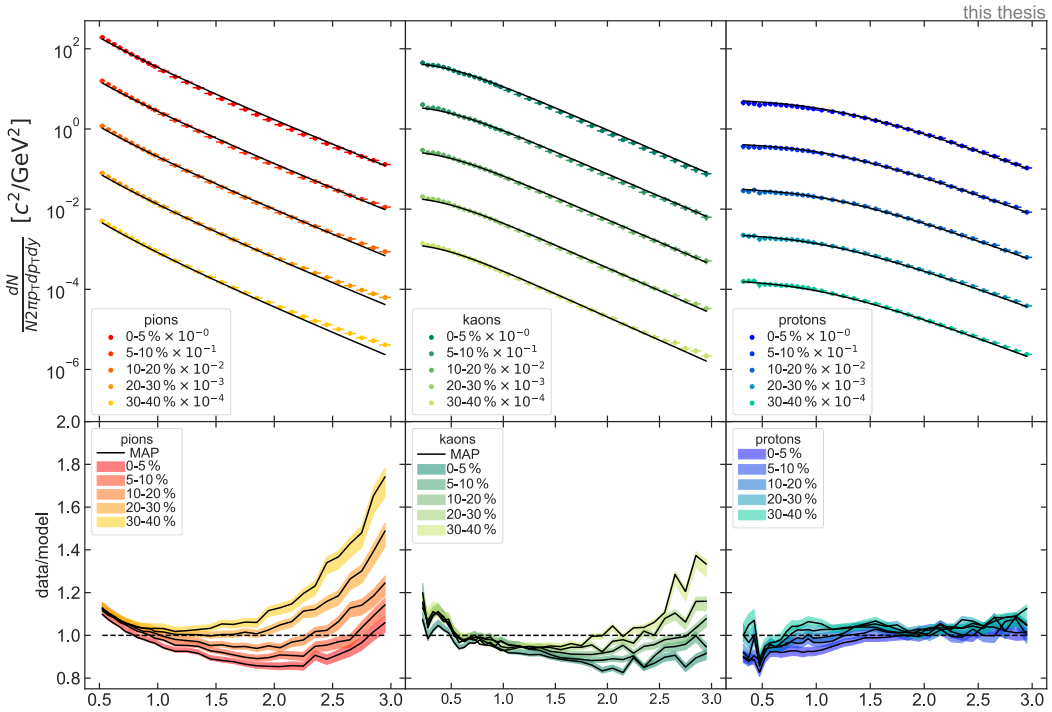


FIGURE 6.1: Top: The MAP estimate ($Norm = 24.9$, $\eta/s = 0.35$, $(\zeta/s)_{\mathrm{max}} = 0.025$, $\tau_0 = 0.10\,\mathrm{fm/c}$, $T_{\mathrm{kin}} = 131.5\,\mathrm{MeV}$, $T_{\mathrm{chem}} = 147.1\,\mathrm{MeV}$) as the best fit to the experimental spectra of pions, kaons, and protons in five centrality classes in Pb-Pb collisions at $\sqrt{s_{\mathrm{NN}}} = 2.76\,\mathrm{TeV}$. Bottom: data-to-model ratios. The error bands were constructed by sampling from the posterior distribution and correspond to the 99 % credible interval.

.

but less pronounced as in the fit to five centrality classes. Therefore, the error of linearization seems to have an effect in driving the thermalization time to lower values, but can not be considered the main reason for it. Further investigations have to show if the usage of more realistic initial conditions with a non-zero fluid velocity can resolve the issue.

The distributions for the kinetic and chemical freeze-out temperatures are both fully captured within the search intervals and follow a Gaussian shape. The kinetic freeze-out temperature $T_{\mathrm{kin}} = 124.5^{+1.9}_{-1.8}\,\mathrm{MeV}$ is significantly different from the fit to five centrality classes ($T_{\mathrm{kin}} = 132.1^{+0.9}_{-1.0}\,\mathrm{MeV}$), whereas the chemical freeze-out temperature $T_{\mathrm{chem}} = 146^{+0.7}_{-0.8}$ is consistent with its previous estimate.

Similarly to the last section, the particle spectra produced with the optimal set of parameters and the simulation model can be compared to the experimental data. The according plots are depicted in fig. 6.2 for the fit of the $0 - 5\,\%$ centrality class.

For all three particles, the simulation mostly agrees with the experimental data within its uncertainties. Only for low and high p_{T} pions ($< 0.75\,\mathrm{GeV/c}$, $> 2.75\,\mathrm{GeV/c}$) and mid-p_{T} kaons ($1\,\mathrm{GeV/c} < p_{\mathrm{T}} < 2.25\,\mathrm{GeV/c}$), significant differences can be observed. It can be expected that the agreement between the data and the model increases, if also the systematic uncertainties of the simulation model are considered. Overall, the experimental data is reproduced with an accuracy of mostly better than 10 % with a reached maximum of 25 % for high-p_{T} pions.

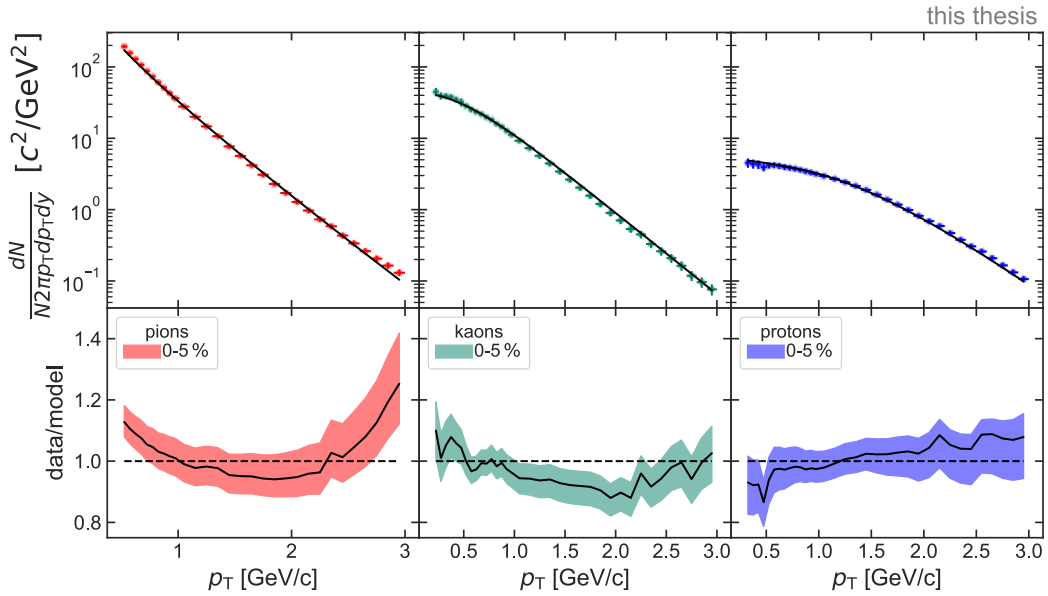


FIGURE 6.2: Top: The MAP estimate ($Norm = 20.7$, $\eta/s = 0.49$, $(\zeta/s)_{\max} = 0.047$, $\tau_0 = 0.12\,\mathrm{fm/c}$, $T_{\mathrm{kin}} = 126.5\,\mathrm{MeV}$, $T_{\mathrm{chem}} = 146.1\,\mathrm{MeV}$) as the best fit to the experimental spectra of pions, kaons, and protons in the centrality class $0-5\,\%$ in Pb-Pb collisions at $\sqrt{s_{\mathrm{NN}}} = 2.76\,\mathrm{TeV}$. Bottom: data-to-model ratios. The error bands were constructed by sampling from the posterior distribution and correspond to the 99 % credible interval.

6.3 NN ensemble model

Because the emulator model is of particular importance in the optimization, it is worthwhile to discuss its choice and performance here.
As already discussed in chapter 5, there are in principle many options for the choice of the emulator model. The de facto standard for computationally expensive simulation models is Gaussian process regression as it provides a flexible regression model with well-calibrated uncertainty quantification. In this thesis, an ensemble of neural networks was used instead to emulate the model combination of Trento, Fluidum, and FastReso. This was done because of the different properties of Fluidum compared to other hydrodynamic simulation models. With very expensive simulation models, only a few hundred parameter configurations can be generated with the simulation in a reasonable time to train the emulator. In these cases, it is most important that the uncertainty measure is good since the parameter space is poorly populated and thus the uncertainties are large. For these situations, Gaussian process regression is superior to other methods, also because the computational complexity of $\mathcal{O}(n^3)$ does not play a role for such small data sets.
In contrast to this, for less demanding simulations where much more data points can be computed with the simulation, Gaussian process regression becomes very inefficient or unfeasible to use due to its computational complexity. One of these more efficient models is Fluidum, which was used in this thesis and allows the generation of 10^4 to 10^5 data points in a reasonable time. To cope with this a much more efficient emulator model had to be chosen. Therefore, in this work, a neural network ensemble was used, which has a computational complexity of $\mathcal{O}(n)$. In the following, the performance of the NN ensemble emulator model is discussed, as well as its uncertainty quantification.

In section 5.5, it was shown, that NNs provide a powerful tool for emulation. The NN was able to reproduce the simulation data within an uncertainty of 1%, which is only a fraction of the uncertainty of the experimental data. However, compared to Gaussian process regression, the definition and training are more complex, as a lot of hyperparameters have to be set and optimized.
To estimate the uncertainty of the NN emulator, several NNs were combined in an ensemble model, and the spread of their predictions was used to estimate the model uncertainty. In particular, as the estimate, the error of the mean was used. This is different from other uses of NN ensembles, where mostly the standard deviation of the predictions is utilized as the uncertainty measure. The error of the mean is more precise but harder to compute, as the correlations between the NNs have to be taken into account. To include them, simplifying assumptions about the correlations between the NNs were made, namely that the mean correlation between two networks is independent of the position in the parameter space. These have been shown to be approximately correct, such that the estimated uncertainty had a good agreement with the true uncertainty obtained from comparing the emulator with the simulation. However, because the optimization procedure was employed in an iterative fashion, the model uncertainty could be decreased until it was vanishingly small compared to the experimental one, such that meaningful parameter estimates could be inferred nonetheless.
Moreover, in this work, two variations of the NN ensemble were tested. One ensemble model has fitted the particle spectra of the simulation, and one the χ^2-value between the simulation and experimental particle spectra. Both models were used to compute the same quantity, the posterior probability of the model parameters, such that both are expected to converge to the same posterior distributions. This was confirmed in this study. However, the spectra-ensemble is recommended over the χ^2-ensemble, since the training process of the NNs is easier, the model is more flexible with respect to including or excluding output bins, and the uncertainty is treated in a straightforward way. The advantage of the χ^2-ensemble, that its estimate of the posterior probability is more accurate because only the uncertainty of one NN node is considered, can be compensated for by the iterative procedure when using the spectra-ensemble.
Overall, the NN ensemble model has been proven to provide reliable and accurate emulation for the simulation of Trento, Fluidum, and FastReso. For this efficient simulation, a NN ensemble model is a superior choice, as the larger data sets can be processed more efficiently than with Gaussian process regression. In addition, the uncertainty estimates have been demonstrated to describe the real uncertainties reliably.

Chapter 7

Conclusion and outlook

In this thesis, estimates of fundamental characteristics of the evolution of heavy-ion collisions were inferred by conducting a Bayesian parameter estimation. For that, the transverse momentum spectra of pions, kaons, and protons from Pb-Pb collisions at $\sqrt{s_{\mathrm{NN}}} = 2.76\,\mathrm{TeV}$ were compared to the output of a simulation consisting of Trento [8], Fluidum [7], and FastReso [9]. In particular, the optimization was performed with respect to the following parameters of the simulation: the normalization $Norm$, the shear viscosity over entropy density η/s, the maximum bulk viscosity over entropy density $(\zeta/s)_{\mathrm{max}}$, the thermalization time τ_0, the kinetic freeze-out temperature T_{kin}, and the chemical freeze-out temperature T_{chem}. For the Bayesian analysis, a procedure was developed ad hoc, which included generating a parameter grid, running the simulation, training an emulator model, and running an MCMC simulation. In contrast to other parameter estimation studies, a neural network ensemble model was used as the emulator model. It has been demonstrated, that neural networks provide a powerful and efficient alternative to Gaussian process regression, which especially may be used if a lot of data can be generated with the simulation. Additionally, it has been shown that their uncertainty can be quantified well within an ensemble model.
The inferred posterior estimates of the parameters are given in chapter 6. They mostly agree with other studies. However, these estimates still have some limitations because of two reasons: Firstly, the systematic uncertainties of the simulation model are not included in the parameter estimates, and secondly, the thermalization time converges to zero, which is incompatible with the physical expectation. The latter is expected to originate from the assumption of zero fluid velocity at the start of the fluid dynamic evolution. To get more realistic initial conditions, the pre-equilibrium phase may be modeled explicitly. For this, work is ongoing to describe this phase by a free-streaming model which eventually will drive the thermalization time to more physical values. Also for the systematic uncertainty work is done, this can be estimated by varying the settings of the fit procedure, such that the optimization is employed for example for different particles, p_{T}-regions or collision systems. For that, already a framework was built such that the analysis can be performed efficiently.
Both of the stated issues are only minor obstacles before meaningful estimates can be obtained, as the general procedure for the optimization remains the same. Therefore the real value of this work lies in the full development of the optimization procedure, including the parameter grid, the neural network ensemble, and the MCMC simulation to infer posterior Bayesian estimates for the parameters. Especially the usage of the neural network ensemble and its uncertainty quantification is a completely new approach in the field of fluid dynamic modeling in heavy-ion collisions. It was specifically tailored to the requirements of the efficient simulation model of Trento, Fluidum, and FastReso, and has been shown to work reliably and accurately as an emulator model.
The presented procedure provides a flexible and powerful method for parameter estimation also in different settings, as it is not bound to the specific use case but is generally applicable. It lays the foundation for a large variety of different analyses in the future. For example, on the basis of this work, additional parameters could be included in the optimization. A natural choice would be the initial state parameters of the Trento

model, which are expected to have a significant impact on the performance of the fit. Moreover, also other physical observables like flow coefficients could be used to better constrain the posterior parameter estimates. Thus, this work only marks the beginning of a variety of further studies that are yet to come in the future.

Bibliography

[1] B. B. Abelev *et al.*, ALICE Collaboration, "Performance of the ALICE Experiment at the CERN LHC," *Int. J. Mod. Phys. A*, vol. 29, p. 1 430 044, 2014. DOI: 10.1142/S0217751X14300440. arXiv: 1402.4476 [nucl-ex].

[2] K. Aamodt, B. Abelev, A. Abrahantes Quintana, *et al.*, ALICE Collaboration, "Higher harmonic anisotropic flow measurements of charged particles in pb-pb collisions at $\sqrt{s_{NN}} = 2.76$ TeV," *Phys. Rev. Lett.*, vol. 107, p. 032 301, 3 2011. DOI: 10.1103/PhysRevLett.107.032301. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.107.032301.

[3] S. Chatrchyan, V. Khachatryan, A. M. Sirunyan, *et al.*, CMS Collaboration, "Measurement of the elliptic anisotropy of charged particles produced in pbpb collisions at $\sqrt{s_{NN}} = 2.76$ tev," *Phys. Rev. C*, vol. 87, p. 014 902, 1 2013. DOI: 10.1103/PhysRevC.87.014902. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevC.87.014902.

[4] G. Aad, B. Abbott, J. Abdallah, *et al.*, ATLAS Collaboration, "Measurement of the azimuthal anisotropy for charged particle production in $\sqrt{s_{NN}} = 2.76$ tev lead-lead collisions with the atlas detector," *Phys. Rev. C*, vol. 86, p. 014 907, 1 2012. DOI: 10.1103/PhysRevC.86.014907. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevC.86.014907.

[5] J. E. Bernhard, "Bayesian parameter estimation for relativistic heavy-ion collisions," Ph.D. dissertation, Duke U., Apr. 2018. arXiv: 1804.06469 [nucl-th].

[6] J. E. Bernhard, J. S. Moreland, and S. A. Bass, "Bayesian estimation of the specific shear and bulk viscosity of quark–gluon plasma," *Nature Phys.*, vol. 15, no. 11, pp. 1113–1117, 2019. DOI: 10.1038/s41567-019-0611-8.

[7] S. Floerchinger, E. Grossi, and J. Lion, "Fluid dynamics of heavy ion collisions with mode expansion," *Phys. Rev. C*, vol. 100, no. 1, p. 014 905, 2019. DOI: 10.1103/PhysRevC.100.014905. arXiv: 1811.01870 [nucl-th].

[8] J. S. Moreland, J. E. Bernhard, and S. A. Bass, "Alternative ansatz to wounded nucleon and binary collision scaling in high-energy nuclear collisions," *Phys. Rev. C*, vol. 92, no. 1, p. 011 901, 2015. DOI: 10.1103/PhysRevC.92.011901. arXiv: 1412.4708 [nucl-th].

[9] A. Mazeliauskas, S. Floerchinger, E. Grossi, and D. Teaney, "Fast resonance decays in nuclear collisions," *Eur. Phys. J. C*, vol. 79, no. 3, p. 284, 2019. DOI: 10.1140/epjc/s10052-019-6791-7. arXiv: 1809.11049 [nucl-th].

[10] Wikipedia. "Standard model." (2019), [Online]. Available: https://en.wikipedia.org/wiki/Standard_Model (visited on 01/14/2022).

[11] M. Thomson, *Modern Particle Physics*. Cambridge: Cambridge University Press, 2013. DOI: 10.1017/CBO9781139525367.

[12] W. Hollik, *Quantum field theory and the standard model*, 2010. arXiv: 1012.3883 [hep-ph].

[13] S. Durr *et al.*, "Ab-Initio Determination of Light Hadron Masses," *Science*, vol. 322, pp. 1224–1227, 2008. DOI: 10.1126/science.1163233. arXiv: 0906.3599 [hep-lat].

[14] P. Zyla *et al.*, "Review of Particle Physics," *PTEP*, vol. 2020, no. 8, p. 083C01, 2020. DOI: 10.1093/ptep/ptaa104.

[15] H. Caines, "The search for critical behavior and other features of the QCD phase diagram – current status and future prospects," *Nuclear Physics A*, vol. 967, pp. 121–128, 2017, The 26th International Conference on Ultra-relativistic Nucleus-Nucleus Collisions: Quark Matter 2017, ISSN: 0375-9474. DOI: https://doi.org/10.1016/j.nuclphysa.2017.05.116. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0375947417302580.

[16] H.-T. Ding, F. Karsch, and S. Mukherjee, "Thermodynamics of strong-interaction matter from Lattice QCD," *Int. J. Mod. Phys. E*, vol. 24, no. 10, p. 1 530 007, 2015. DOI: 10.1142/S0218301315300076. arXiv: 1504.05274 [hep-lat].

[17] P. Steinbrecher, "The QCD crossover at zero and non-zero baryon densities from Lattice QCD," *Nucl. Phys. A*, vol. 982, F. Antinori, A. Dainese, P. Giubellino, V. Greco, M. P. Lombardo, and E. Scomparin, Eds., pp. 847–850, 2019. DOI: 10.1016/j.nuclphysa.2018.08.025. arXiv: 1807.05607 [hep-lat].

[18] M. G. Alford, A. Schmitt, K. Rajagopal, and T. Schäfer, "Color superconductivity in dense quark matter," *Rev. Mod. Phys.*, vol. 80, pp. 1455–1515, 2008. DOI: 10.1103/RevModPhys.80.1455. arXiv: 0709.4635 [hep-ph].

[19] S. Sarkar, H. Satz, and B. Sinha, *The physics of the Quark-Gluon Plasma: Introductory Lectures*. Berlin Heidelberg: Springer, 2010. DOI: 10.1007/978-3-642-02286-9.

[20] P. Braun-Munzinger and B. Dönigus, "Loosely-bound objects produced in nuclear collisions at the LHC," *Nuclear Physics A*, vol. 987, Apr. 2019. DOI: 10.1016/j.nuclphysa.2019.02.006.

[21] A. Jaiswal and V. Roy, "Relativistic hydrodynamics in heavy-ion collisions: General aspects and recent developments," *Advances in High Energy Physics*, vol. 2016, pp. 1–39, Jan. 2016. DOI: 10.1155/2016/9623034.

[22] P. Braun-Munzinger, J. Stachel, and C. Wetterich, "Chemical freezeout and the QCD phase transition temperature," *Phys. Lett. B*, vol. 596, pp. 61–69, 2004. DOI: 10.1016/j.physletb.2004.05.081. arXiv: nucl-th/0311005.

[23] J. E. Bernhard, J. S. Moreland, S. A. Bass, J. Liu, and U. Heinz, "Applying Bayesian parameter estimation to relativistic heavy-ion collisions: simultaneous characterization of the initial state and quark-gluon plasma medium," *Phys. Rev. C*, vol. 94, no. 2, p. 024 907, 2016. DOI: 10.1103/PhysRevC.94.024907. arXiv: 1605.03954 [nucl-th].

[24] D. Devetak, A. Dubla, S. Floerchinger, *et al.*, "Global fluid fits to identified particle transverse momentum spectra from heavy-ion collisions at the Large Hadron Collider," *JHEP*, vol. 06, p. 044, 2020. DOI: 10.1007/JHEP06(2020)044. arXiv: 1909.10485 [hep-ph].

[25] C. Gale, S. Jeon, and B. Schenke, "Hydrodynamic Modeling of Heavy-Ion Collisions," *Int. J. Mod. Phys. A*, vol. 28, p. 1 340 011, 2013. DOI: 10.1142/S0217751X13400113. arXiv: 1301.5893 [nucl-th].

[26] M. L. Miller, K. Reygers, S. J. Sanders, and P. Steinberg, "Glauber modeling in high-energy nuclear collisions," *Annual Review of Nuclear and Particle Science*, vol. 57, no. 1, pp. 205–243, 2007. DOI: 10.1146/annurev.nucl.57.090506.123020. eprint: https://doi.org/10.1146/annurev.nucl.57.090506.123020. [Online]. Available: https://doi.org/10.1146/annurev.nucl.57.090506.123020.

[27] A. Jaiswal and V. Roy, "Relativistic hydrodynamics in heavy-ion collisions: General aspects and recent developments," *Advances in High Energy Physics*, vol. 2016, 1–39, 2016, ISSN: 1687-7365. DOI: 10.1155/2016/9623034. [Online]. Available: http://dx.doi.org/10.1155/2016/9623034.

[28] P. F. Kolb and U. W. Heinz, "Hydrodynamic description of ultrarelativistic heavy ion collisions," R. C. Hwa and X.-N. Wang, Eds., pp. 634–714, May 2003. arXiv: nucl-th/0305084.

[29] J. Casalderrey-Solana, H. Liu, D. Mateos, K. Rajagopal, and U. A. Wiedemann, *Gauge/String Duality, Hot QCD and Heavy Ion Collisions*. Cambridge University Press, 2014, ISBN: 978-1-139-13674-7. DOI: 10.1017/CBO9781139136747. arXiv: 1101.0618 [hep-th].

[30] W. A. Hiscock and L. Lindblom, "Generic instabilities in first-order dissipative relativistic fluid theories," *Phys. Rev. D*, vol. 31, pp. 725–733, 4 1985. DOI: 10.1103/PhysRevD.31.725. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevD.31.725.

[31] S. Floerchinger, *A very brief introduction to heavy ion physics*, Lecture notes for the proceedings of the 2015 European School of High-Energy Physics in Bansko, Bulgaria., 2015.

[32] P. Kovtun, D. T. Son, and A. O. Starinets, "Viscosity in strongly interacting quantum field theories from black hole physics," *Phys. Rev. Lett.*, vol. 94, p. 111 601, 2005. DOI: 10.1103/PhysRevLett.94.111601. arXiv: hep-th/0405231.

[33] F. Cooper and G. Frye, "Single-particle distribution in the hydrodynamic and statistical thermodynamic models of multiparticle production," *Phys. Rev. D*, vol. 10, pp. 186–189, 1 1974. DOI: 10.1103/PhysRevD.10.186. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevD.10.186.

[34] G. Torrieri, S. Steinke, W. Broniowski, W. Florkowski, J. Letessier, and J. Rafelski, "SHARE: Statistical hadronization with resonances," *Comput. Phys. Commun.*, vol. 167, pp. 229–251, 2005. DOI: 10.1016/j.cpc.2005.01.004. arXiv: nucl-th/0404083.

[35] J. Sollfrank, P. Koch, and U. Heinz, "Is there a low-pT"anomaly" in the pion momentum spectra from relativistic nuclear collisions?" *Zeitschrift für Physik C Particles and Fields*, vol. 52, no. 593, pp. 1431–5858, 1991. DOI: 10.1007/BF01562334.

[36] H. Bebie, P. Gerber, J. Goity, and H. Leutwyler, "The role of the entropy in an expanding hadronic gas," *Nuclear Physics B*, vol. 378, no. 1, pp. 95–128, 1992, ISSN: 0550-3213. DOI: https://doi.org/10.1016/0550-3213(92)90005-V. [Online]. Available: https://www.sciencedirect.com/science/article/pii/055032139290005V.

[37] F. Emmert-Streib, Z. Yang, H. Feng, S. Tripathi, and M. Dehmer, "An introductory review of deep learning for prediction models with big data," *Frontiers in Artificial Intelligence*, vol. 3, 2020, ISSN: 2624-8212. DOI: 10.3389/frai.2020.00004. [Online]. Available: https://www.frontiersin.org/article/10.3389/frai.2020.00004.

[38] J. Schmidhuber, "Deep learning in neural networks: An overview," *CoRR*, vol. abs/1404.7828, 2014. arXiv: 1404.7828. [Online]. Available: http://arxiv.org/abs/1404.7828.

[39] D. Rumelhart, G. Hinton, and R. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 9, 1986.

[40] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 2007, ISBN: 0387310738.

[41] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," Proceedings of Machine Learning Research, vol. 9, Y. W. Teh and M. Titterington, Eds., pp. 249–256, 2010. [Online]. Available: https://proceedings.mlr.press/v9/glorot10a.html.

[42] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. 61, pp. 2121–2159, 2011. [Online]. Available: http://jmlr.org/papers/v12/duchi11a.html.

[43] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014. arXiv: 1412.6980.

[44] J. Gawlikowski, C. R. N. Tassi, M. Ali, *et al.*, "A survey of uncertainty in deep neural networks," *arXiv preprint arXiv:2107.03342*, 2021.

[45] O. Sagi and L. Rokach, "Ensemble learning: A survey," *WIREs Data Mining and Knowledge Discovery*, vol. 8, no. 4, e1249, 2018. DOI: https://doi.org/10.1002/widm.1249. [Online]. Available: https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/widm.1249.

[46] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," vol. 30, I. Guyon, U. V. Luxburg, S. Bengio, *et al.*, Eds., 2017. [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/9ef2ed4b7fd2c810847ffa5fa85bce38-Paper.pdf.

[47] S. Chib and E. Greenberg, "Understanding the metropolis-hastings algorithm," *The American Statistician*, vol. 49, no. 4, pp. 327–335, 1995, ISSN: 00031305. [Online]. Available: http://www.jstor.org/stable/2684568 (visited on 04/28/2022).

[48] G. Casella and E. I. George, "Explaining the gibbs sampler," *The American Statistician*, vol. 46, no. 3, pp. 167–174, 1992, ISSN: 00031305. [Online]. Available: http://www.jstor.org/stable/2685208 (visited on 04/28/2022).

[49] R. M. Neal *et al.*, "Mcmc using hamiltonian dynamics," *Handbook of markov chain monte carlo*, vol. 2, no. 11, p. 2, 2011. arXiv: 1206.1901.

[50] A. D. Sokal, "Monte carlo methods in statistical mechanics: Foundations and new algorithms note to the reader," 1996.

[51] J. S. Moreland, J. E. Bernhard, and S. A. Bass, "Bayesian calibration of a hybrid nuclear collision model using p-Pb and Pb-Pb data at energies available at the CERN Large Hadron Collider," *Phys. Rev. C*, vol. 101, no. 2, p. 024 911, 2020. DOI: 10.1103/PhysRevC.101.024911. arXiv: 1808.02106 [nucl-th].

[52] B. Abelev *et al.*, "Centrality dependence of π, K, p production in Pb-Pb collisions at $\sqrt{s_{NN}}$ = 2.76 TeV," *Phys. Rev. C*, vol. 88, p. 044 910, 2013. DOI: 10.1103/PhysRevC.88.044910. arXiv: 1303.0737 [hep-ex].

[53] E. Schulz, M. Speekenbrink, and A. Krause, "A tutorial on gaussian process regression: Modelling, exploring, and exploiting functions," *Journal of Mathematical Psychology*, vol. 85, pp. 1–16, 2018, ISSN: 0022-2496. DOI: https://doi.org/10.1016/j.jmp.2018.03.001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0022249617302158.

[54] R. M. Neal, "Priors for infinite networks," 1996. [Online]. Available: https://www.cs.toronto.edu/~radford/ftp/pin.pdf.

[55] A. Paszke, S. Gross, F. Massa, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[56] S. S. Shapiro and M. B. Wilk, "An analysis of variance test for normality (complete samples)," *Biometrika*, vol. 52, no. 3/4, pp. 591–611, 1965, ISSN: 00063444. [Online]. Available: http://www.jstor.org/stable/2333709 (visited on 08/23/2022).

[57] D. Foreman-Mackey, D. W. Hogg, D. Lang, and J. Goodman, "Emcee: The mcmc hammer," *PASP*, vol. 125, pp. 306–312, 2013. DOI: 10.1086/670067. eprint: 1202.3665.

[58] J. Goodman and J. Weare, "Ensemble samplers with affine invariance," *Communications in Applied Mathematics and Computational Science*, vol. 5, no. 1, pp. 65–80, Jan. 2010. DOI: 10.2140/camcos.2010.5.65.

[59] S. Ryu, J. F. Paquet, C. Shen, *et al.*, "Importance of the Bulk Viscosity of QCD in Ultrarelativistic Heavy-Ion Collisions," *Phys. Rev. Lett.*, vol. 115, no. 13, p. 132 301, 2015. DOI: 10.1103/PhysRevLett.115.132301. arXiv: 1502.01675 [nucl-th].

[60] F. A. Flor, G. Olinger, and R. Bellwied, "Flavour and energy dependence of chemical freeze-out temperatures in relativistic heavy ion collisions from rhic-bes to lhc energies," *Physics Letters B*, vol. 814, p. 136 098, 2021, ISSN: 0370-2693. DOI: https://doi.org/10.1016/j.physletb.2021.136098. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0370269321000381.