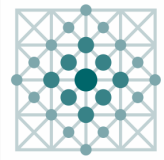


Applied and Computational Mathematics

RWTH AACHEN UNIVERSITY

This work was submitted to the
Diese Arbeit wurde vorlegegt am
Research Lab for Applied and Computational Mathematics (ACoM)

Deterministic Reconstruction in Electron Probe Microanalysis using a Flexible Material Parametrization

Master Thesis, RWTH Aachen University
Computational Engineering Science
November 2021

presented by

TAMME CLAUS, B.SC.
RWTH Aachen University
Mat.Nr: 346980
tamme.claus@rwth-aachen.de

supervised by

MANUEL TORRILHON, PROF. DR.
Research Lab for Applied and Computational Mathematics, ACoM
RWTH Aachen University
mt@acom.rwth-aachen.de

JONAS BÜNGER, DR.
Research Lab for Applied and Computational Mathematics, ACoM
RWTH Aachen University
buenger@acom.rwth-aachen.de

Preface

This thesis marks the end of my master study in Computational Engineering Science at RWTH Aachen University. A journey that led me to research on electron probe microanalysis as early as the winter term of 2016 which resulted in numerous smaller seminar projects, a bachelor thesis "Application of the Adjoint Method in Gradient-based Optimization to the M1-Model in Electron Beam Microanalysis" and a publication "A Novel Reconstruction Method to Increase Spatial Resolution in Electron Probe Microanalysis".

I am very grateful to my advisor and professor, Dr. Manuel Torrilhon, who introduced me to the topic of reconstruction in EPMA and allowed me to continue this research. You were always open-minded about any digressions I took into various other methods and frameworks I considered useful for our problem. One of which led to the writing of this thesis.

Sincere thanks also to my advisor, Dr. Jonas Bünger. While you accompanied me through the longest part of my research, I learned a lot from you about mathematical methods, scientific writing, but also about the endurance needed to succeed in academic research.

Additionally, I would like to thank Dr. Silvia Richter for sharing her expertise about the experimental side of EPMA in inspiring discussions. Also, Dr. Uwe Naumann for giving me the chance for a deep dive into Algorithmic Differentiation during a seminar. And Dr. Raul Tempone for offering me the possibility to analyze the statistical nature of the inverse problem in EPMA in the form of a project alongside his lecture.

I also thank my friends and fellow students and especially my former flatmate, André Stawarski, for making my student life a time to look back on with pleasure and for at least trying to understand explanations about my research. Of course also for proofreading parts of this thesis.

For giving me the opportunity and at all times providing me the necessary support to study what I want, I am deeply grateful to my family. My thoughts are also with my uncle, Bernd Claus, who unexpectedly passed away while I was working on this thesis. Judging from [Claus et al., 2017], you were probably the only person in the close family with whom I could have discussed deeper parts of my research. I am grateful for the time we spent together despite the local distance, remembering especially our lively discussion of $14, 19, 28, 47, 61$ and 75[1].

Tamme Claus, November 2021

[1] These are the 2-digit Keith numbers, e.g. 14: $1 + 4 = 5$, $4 + 5 = 9$, $5 + 9 = 14$. Finding longer-digit Keith numbers is a computational challenge.

Abstract

Electron Probe Microanalysis (EPMA) is a non-destructive technique to determine the chemical composition of material samples in the micro- to nanometer range. Based on intensity measurements of characteristic x-radiation, information about the chemical composition of the sample is obtained. The determination of underlying chemical composition represents the inverse problem of reconstruction in EPMA.

All currently applied reconstruction methods are based on models assuming a homogeneous or layered structure of the reconstructed material. To increase the spatial resolution of reconstruction in EPMA the combination of a more sophisticated reconstruction method, that is based on a model which allows complex material structure, together with multiple measurements with varying beam configurations is required. The diversity of the application fields of EPMA poses a challenge for reconstruction methods. The method should be able to answer different questions on material structure and take into account different prior knowledge about material structure.

We present a reconstruction method based on the combination of gradient-based optimization methods and a deterministic k-ratio model built on the P_N model, an approximation of the linear Boltzmann equation. In addition, we present an efficient and extensible method for differentiation of our model, which equips the reconstruction method with general applicability. By specifying a material parametrization, the method can be adapted to incorporate prior knowledge and reconstruct a wide variety of material structures. The core of the reconstruction method is the computation of the gradient (the differentiation method) which relies on a combination of the 'adjoint state method' and the 'adjoint mode' of algorithmic differentiation. Through examples, the flexibility of the k-ratio model and the generality of the reconstruction/differentiation method is demonstrated.

Zusammenfassung

Deterministische Rekonstruktion in Elektronenstrahl-Mikroanalyse unter Verwendung einer flexiblen Materialparametrisierung

Die Elektronenstrahl-Mikroanalyse (engl. electron probe microanalysis, EPMA) ist eine zerstörungsfreie Technik zur Bestimmung der chemischen Zusammensetzung von Materialproben im Mikro- bis Nanometerbereich. Auf der Grundlage von Intensitätsmessungen der charakteristischen Röntgenstrahlung werden Informationen über die chemische Zusammensetzung der Probe gewonnen. Die Ermittlung der zugrunde liegenden chemischen Zusammensetzung stellt das inverse Problem der Rekonstruktion in EPMA dar.

Alle derzeit angewandten Rekonstruktionsmethoden basieren auf Modellen, die von einer homogenen oder geschichteten Struktur des rekonstruierten Materials ausgehen. Um die räumliche Auflösung der Rekonstruktion in EPMA zu erhöhen, ist die Kombination mehrerer Messungen mit unterschiedlichen Strahlkonfigurationen zusammen mit einer neuartigen Rekonstruktionsmethode erforderlich. Die Vielfalt der Anwendungsbereiche der EPMA stellt eine Herausforderung für Rekonstruktionsmethoden dar. Die Methode sollte in der Lage sein, verschiedene Fragestellungen zu beantworten und unterschiedliches Vorwissen über die Materialstruktur miteinzubeziehen.

Wir stellen eine Rekonstruktionsmethode vor, die auf der Kombination von Gradienten-basierten Optimierungsmethoden und einem deterministischen k-ratio-Modell basiert, das auf dem P_N-Modell, einer Approximation der linearen Boltzmann Gleichung, aufbaut. Zusätzlich stellen wir eine effiziente und erweiterbare Methode zur Differenzierung unseres Modells vor, die die Rekonstruktionsmethode mit allgemeiner Anwendbarkeit ausstattet. Durch die Spezifikation einer Materialparametrisierung kann die Methode so angepasst werden, dass sie Vorwissen miteinbezieht und verschiedenste Materialstrukturen rekonstruieren kann. Den Kern der Rekonstruktionsmethode bildet die Berechnung des Gradienten (die Differenzierungsmethode), die auf eine Kombination der 'adjoint state method' und dem 'adjoint mode' der algorithmischen Differenzierung setzt. Anhand von Beispielen wird die Flexibilität des k-ratio-Modells und die allgemeine Anwendbarkeit der Rekonstruktions-/Differenzierungsmethode demonstriert.

Contents

Introduction

1 The Inverse Problem of Reconstruction in EPMA

Electron Probe Microanalysis (EPMA) [Reimer, 1998; Heinrich and Newbury, 1991] is an imaging technique used for the quantitative analysis of the composition of solid material samples at the micro- to nanometer scale. The sample is excited by a focussed beam of electrons which induces multiple relaxation processes inside the sample. In EPMA the emission of characteristic x-rays is of special focus. If an electron which is induced by the beam strikes a bound electron which occupies an atomic shell of an atom inside the specimen, the bound electron is ejected from its shell and the atom is left with a vacancy. Outer shell electrons fill this vacancy by emitting a quantized x-ray with an energy corresponding to the energy level difference of the originating and the target shell. The energy levels of electron shells are characteristic for a specific atom, hence the energy of the emitted x-ray provides information about the composition of the material sample. In Figure 1 we outline the main physical processes that occur during an experiment.

In EPMA the emitted x-ray intensity is measured by counting detected x-rays. Because of multiple uncertainties concerning the experimental instrument, the x-ray intensity $I_{(Z,j)}$ is normalized into a k-ratio $k_{(Z,j)}$ using standard intensities $I^{\text{std}}_{(Z,j)}$ measured from a known reference sample.

$$k_{(Z,j)} = \frac{I_{(Z,j)}}{I^{\text{std}}_{(Z,j)}} \tag{1}$$

Standard intensities $I^{\text{std}}_{(Z,j)}$ are measured using the same experimental setup (except the sample) as used for $I_{(Z,j)}$, therefore the normalization eliminates uncertain multiplicative factors which influence the intensity, e.g. the detector efficiency. We denote a characteristic x-ray (Z, j) by the atomic number Z of the associated atom and by a transition j. The x-ray transition encodes target and originating shell of the electron transition, where we use IUPAC notation, e.g. $K - L_2$ for a transition from the L_2 shell to the K shell.

An experiment yields multiple k-ratios; one for each of the considered x-ray transition. Additionally, the experimental setup $\boldsymbol{\mathcal{U}}$, e.g. the beam position, energy or angle, can be varied to obtain information about the material. But the collection of all k-ratios $k = \{k^{\mathcal{U}_1}_{(Z,j)}, \ldots, k^{\mathcal{U}_2}_{(Z,j)}, \ldots\}$ does not directly reveal the composition of the material sample and a reconstruction process is necessary. The crucial parts of a reconstruction are the definition of parameters p which describe a material composition and the definition of a k-ratio model $k(p)$ which predicts the observed k-ratios given a material composition p. Then the reconstruction

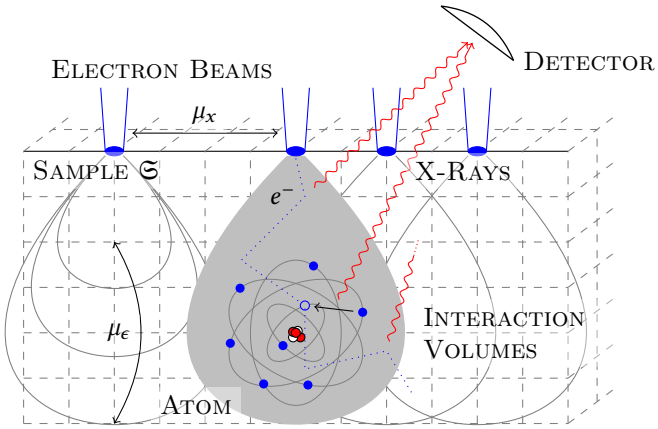


Figure 1: *A sketch of the physical processes in EPMA. The sample $\mathfrak{S}$ is rastered by electron beams (blue) with different position μ_x. Electrons e^- from the electron beam (blue dotted line) scatter inside the sample and strike bound electrons which leave a vacancy (blue circle). Outer shell electrons (blue discs) fill this vacancy and release an x-ray of characteristic energy (red wobbly line). The x-ray travels through the sample and is counted by a detector. Beam electrons only excite a certain volume of the sample, the interaction volume (gray ellipses) which scales with the beam energy μ_ϵ.*

can be formalized as follows:

$$p^* = \arg\min_p \operatorname{disc}(k(p), k^{\exp}). \tag{2}$$

Which means: find the set of parameters p^* which minimize a discrepancy $\operatorname{disc}(\cdot,\cdot)$ of modelled k-ratios $k(p)$ and experimental k-ratios $k^{\exp}$. The definition of the parameters p and the model $k(p)$ are decisive for the question(s) posed to the experiment, as explained in the following.

2 Multifaceted Nature of the Reconstruction Problem

The applications of EPMA are manifold [Pinard et al., 2013; Moy et al., 2019; Llovet et al., 2021]. From geology, and material science to electronics, researchers rely on EPMA to determine material composition and structure of samples. Prior to the analysis the knowledge about the material structure is different for each application. Whereas a material scientist investigates e.g. a coated material with known constituents but unknown thicknesses, a geologist might be interested in the chemical composition of a material.

Simultaneously, diverse applications pose a wide variety of questions for the reconstruction to unveil. A basic question is the determination of homogeneous concentrations inside the sample. A more advanced investigation is the reconstruction of structure (e.g. the identification of the thickness of coatings, the determination of position and size of inclusions or the shape characterization of diffusive material interfaces). The most general question is the reconstruction of a general description of compounds. In this work we consider mass concentrations $\rho(x) : \mathbb{R}^3 \to \mathbb{R}^{n_e}$ as the general description of compounds (n_e is the number of constituents). Hence, the general reconstruction problem is

$\rho^*(x) = \arg\min_{\rho(x)} \mathrm{disc}(k(\rho(x)), k^{\mathrm{exp}})$.

The unique reconstruction of infinite dimensional mass concentrations $\rho^*(x)$ from a finite set of measurements k^{exp} is only possible with additional constraints or prior assumptions. Additionally, the precise representation of uncertainty in $\rho^*(x)$ is a useful approach. In the mathematical framework of inverse problem theory, knowing the uncertainty of the unknown $\rho^*(x)$ is defined as the solution of the inverse reconstruction problem. Prior knowledge about the material and the additional knowledge based on the experimental measurements reduces general ignorance of $\rho^*(x)$.

2.1 Inverse Problem Theory

Inverse problem theory [Tarantola, 2005; Stuart, 2010] bases on the assumption, that the observation (the k-ratio measurements k^{exp}) relate with the model parameters (the material description ρ) through

$$k^{\mathrm{exp}} = k(\rho) + \eta, \tag{3}$$

where η is a random variable. More generally, one defines the likelihood, the probability that based on the material description ρ the model $k(\rho)$ produces the data k^{exp}. The probability density function (pdf) of the likelihood is

$$\pi(k^{\mathrm{exp}}|\rho) = \pi_\eta(k^{\mathrm{exp}} - k(\rho)), \tag{4}$$

where π_η is the pdf of the random variable η, which is usually referred to as noise, but might also include model uncertainties.

Additionally, the prior knowledge about the material is encoded the prior pdf $\pi(\rho)$. Using Bayes Theorem, the resulting uncertainty about the material is the posterior pdf

$$\pi(\rho|k^{\mathrm{exp}}) = \frac{\pi(k^{\mathrm{exp}}|\rho)\pi(\rho)}{\int \pi(k^{\mathrm{exp}}|\rho)\pi(\rho)\,\mathrm{d}\rho}. \tag{5}$$

The posterior is the probability, that ρ represents the reality given the measured observation k^{exp}. The posterior $\pi(\rho|k^{\mathrm{exp}})$ combines likelihood $\pi(k^{\mathrm{exp}}|\rho)$ and prior $\pi(\rho)$ and defines the solution of the inverse problem.

The statistical approach to inverse problems is philosophical and for problems of a certain size not (yet) applicable. However, it motivates concepts which are applied to solve real world reconstruction problems. In Section B.1 we mention the computation of maximum likelihood or maximum posterior estimates, which relate to Equations (4) and (5). Also, the specification of the prior $\pi(\rho)$ refers to regularization, a concept we discuss in the next section.

2.2 Regularization

The representation of the unknown $\rho(x)$ as a function has infinite dimensions. Therefore, it can not be reconstructed from a finite set of data k^{exp} even in the absence of noise. The inverse problem is considered as being ill-posed, because no unique solution exists. To mitigate the ill-posed behavior of the inverse problem, regularization adds information. Thus, it is closely connected to the definition of prior knowledge about the material ρ from inverse problem theory.

Regularization is a far-reaching strategy that can be implemented in many forms. Benning and Burger [2018] give a comprehensive overview over classical and modern regularization techniques. Often, they rely on the assumption of a certain regularity of the parameters, concerning e.g. their value or their gradient. A well-studied regularization technique is Tikhonov regularization, which assumes that the unknown is close to some expected value.

We mainly think of regularization as the definition of a parametrization of the material description using $\rho(x; p)$. Instead of specifying the material as a function $\rho(x)$ with infinite dimensions, we reduce the dimensionality to a finite dimensional parameter vector p, which parametrizes the mass concentrations $\rho(x; p)$. The parametrization translates the general reconstruction problem, which searches for mass concentrations $\rho^*(x)$, into the reconstruction problem given in Equation (2), which searches for parameters p^* that describe structure in the material.

Reducing the flexibility of the unknown $\rho(x) \rightsquigarrow \rho(x; p)$ must be accompanied by the addition of carefully chosen material knowledge, because the specification of a parametrization predefines all possible reconstruction results. Furthermore, the parametrization should fit the available measurements. The more flexible the parametrization is, the more measurements have to be provided, otherwise the measurements lack enough information and the reconstruction result is ambiguous.

Examples A researcher might be interested in the thicknesses of multiple coatings, but knows the composition of each coating. He encodes his prior assumptions into a material parametrization $\rho(x; p)$, provides measurements k^{exp} and submits the task of reconstructing the thicknesses p^*.

Another research might be interested in the size, position and composition of a circular material inclusion, but knows the substrate composition. He also specifies his prior knowledge (substrate composition and inclusion shape) to the reconstruction problem in the form of a material parametrization $\rho(x; p)$. The computation of the material parameters p^* is then subject to the reconstruction method.

Similarly, many other tasks can be specified up to the case where the researcher does not know anything about the material a priori. Then he would iterate and experiment with multiple material parametrizations until he is convinced, that the material model sufficiently explains his available data.

2.3 Classical K-Ratio Models and Reconstruction Methods

Computational models, which are currently being used to predict k-ratios, can be classified in two categories: Monte Carlo models and ZAF/$\phi(\rho z)$-models

For inhomogeneous materials Monte Carlo models [Llovet and Salvat, 2017; Ritchie, 2009] are utilized. By sampling of electron trajectories based on random scattering processes in the material, Monte Carlo models approximate the electron density distribution which is induced by the bombardment with beam electrons. From the electron density distribution, k-ratios can be computed by accounting for ionization, fluorescence and absorption effects. However, Monte Carlo models are subject to statistical noise and are only deterministic in the limit (by sampling infinitely many electron trajectories). The statistical noise hinders the application of k-ratio models based on Monte Carlo methods in a reconstruction. For a

reconstruction, the modelled k-ratios must be repeatedly evaluated and compared for similar materials. If the k-ratios are superimposed by noise, no clear comparison is possible.

For homogeneous samples or samples which have a layered structure in depth, ZAF and $\phi(\rho z)$-models [Pouchou and Pichoir, 1991; Heinrich and Newbury, 1991; Riveros and Castellano, 1993], which directly approximate the ionization distribution have been developed. While being very restrictive on the material structure, ZAF/$\phi(\rho z)$-models are very efficient in computation. Reconstruction methods (termed Matrix Correction methods) which are currently being used mainly build on ZAF/$\phi(\rho z)$-models. Therefore, the reconstruction methods and their results are also constrained by the assumption of homogeneity or a layered structure.

The constraint of the material structure by the k-ratio model underlying the reconstruction method currently defines the analytical spatial resolution of EPMA [Moy and Fournelle, 2017; Buse and Kearns, 2020; Carpenter and Jolliff, 2015]. The interaction volume, the part of the sample in which x-rays are generated is assumed to be homogeneous, and the sample is rastered by an electron beam ensuring that the interaction volumes of the different experiments do not overlap to avoid ambiguous results. The material composition is reconstructed independently for each electron beam position and the reconstruction result is associated with the homogeneous material composition of the respective interaction volume. Approaches to increase the spatial resolution based on the physical reduction of the interaction volume have been proposed [Moy et al., 2019]. However, smaller interaction volumes also mean a weaker x-ray intensity and thus stronger noise in the measurement data.

We propose to apply more sophisticated reconstruction methods, which fuse k-ratios measurements with different experimental setups and base on a model which allows inhomogeneous material structures smaller than the size of the interaction volume. Obviously, the softening of the restriction that materials are homogeneous needs more information in the available k-ratio measurements. The fusion of k-ratio measurements with different beam position and different beam energies into one reconstruction process shows promising evidence, that material structures smaller than the interaction volume can be resolved [Richter et al., 2013; Claus et al., 2021].

3 Design of Our Model and Reconstruction Method - Structure of the Thesis

The diversity of questions which are posed to EPMA illustrates the need to develop a reconstruction method which remains general in the material parametrization. We base our model on the same mathematical framework as Monte Carlo models. The electron transport is governed by the Boltzmann equation, however instead of random sampling, we rely on a deterministic approximation of the Boltzmann equation [Larsen et al., 1997; Duclous et al., 2010; Mevenkamp, 2016; Bünger, 2021]. This allows the exact computation of gradients, the main building block of reconstruction methods. Gradients based on Monte Carlo models are polluted by statistical noise and their use in reconstruction methods is restricted.

On top of the k-ratio model, we propose to use a combination of Algorithmic Differentiation [Naumann, 2011; Griewank, 2003] and the Adjoint State Method [Plessix, 2006] to compute gradients of the k-ratio model. Algorithmic Differentiation is a method to com-

pute gradients of numerical functions in a systematic (usually even automatic) and efficient manner. The Adjoint State Method describes a way to efficiently differentiate models based on partial differential equations (our Boltzmann approximation). The combination of both allows us to remain generic both in the objective function and the material parametrization.

Structure of the Thesis In Chapter A (The Forward Problem) we describe the individual parts of our k-ratio model: the P_N-model for electron transport and its numerical approximation using StaRMAP, the combination with ionization and fluorescence effects, and the approximation of x-ray attenuation. We conclude the first chapter with validations of our implementation and showcase its flexibility.

In Chapter B (The Inverse Problem) we motivate the formulation of the reconstruction problem as an optimization problem, briefly introduce objective functions and gradient-based optimization methods and then describe the combination of Algorithmic Differentiation with the Adjoint State Method to compute gradients of our model. Chapter B is concluded with a validation of the gradient computation and artificial reconstruction experiments.

This thesis is finalized with an outlook which summarizes the findings and motivates further research of reconstruction in EPMA using deterministic transport equations.

Implementation of the Described Methods in the Programming Language julia The programming language julia (julia) is a rapidly growing programming language, which is mainly focused on the implementation of numerical methods. While providing a convenient programming syntax, it also enables high computational efficiency due to its just-in-time compiler and the concept of dynamic dispatch. It might also be the language of choice for further development of the reconstruction method, because using the concept of dynamic dispatch, various optimizations of the reconstruction method based on different parametrizations can be implemented conveniently.

As part of this thesis we developed `StaRMAP.jl`, a generic solver of radiation transport equations (closely following the ideas presented in Seibold and Frank [2014] and Bünger [2021]) which is implemented using the programming language julia.

On top of `StaRMAP.jl` we implemented routines, which can be utilized for a convenient setup of numerical experiments to compute k-ratios and all other underlying physical quantities. Thereby we depend on `NeXLCore.jl` [Ritchie, 2021b], a library which collects core algorithms and data for x-ray microanalysis. The computation of the adjoint state equation also utilizes `StaRMAP.jl`.

Additionally, we implemented multiple material parametrizations, which are differentiable and can be linked to Algorithmic Differentiation libraries in julia: e.g. `Zygote.jl` [Innes, 2018a], `ReverseDiff.jl` [Kelley, 2021] and `ForwardDiff.jl` [Revels et al., 2016]. One of the parametrizations uses core ideas of the neural network library `Flux.jl` [Innes, 2018b].

The connection to AD tools is exploited when computing gradients using the adjoint state method. We also link the adjoint state method to the AD libraries, which has the advantage, that both, the parametrization and the objective function, can be replaced effortlessly.

Chapter A

The Forward Problem

With the ultimate goal of solving the inverse problem of reconstruction, we must first specify the forward problem. We dedicate Chapter A to the specification and solution of the forward problem.

Chapter A is divided in Sections A.1 to A.4. In Section A.1 we describe and derive the mathematical formulation of the deterministic k-ratio model, while in Section A.2 we address the numerical computation of the introduced model. Sections A.1 and A.2 remain generic in the material description $\rho(x)$, but we exemplarily present possible parametrization of $\rho(x; p)$ in Section A.3. Section A.4 concludes Chapter A with the demonstration of numerical experiments.

A.1 A K-Ratio Model based on the PN-Model for Electron Transport

In this section we present a deterministic k-ratio model [Mevenkamp, 2016; Bünger, 2021], which is based on the P_N moment expansion of the linear Boltzmann Equation in Continuous Slowing Down approximation (BCSD). We specify the material by its mass concentrations in Section A.1.1 as the input to the forward problem and discuss the successive parts in the computation of k-ratios in Sections A.1.2 to A.1.6. In Section A.1.7 we describe the modeling of the beam.

We aim for a generic model for k-ratios $k_{(Z,j)}$ based on mass concentrations $\rho(x)$

$$k_{(Z,j)}(\rho(x)) = \frac{I_{(Z,j)}}{I^{\mathrm{std}}_{(Z,j)}} = \frac{I_{(Z,j)}(\rho(x))}{I_{(Z,j)}(\rho^{\mathrm{std}}(x))}, \tag{A.1}$$

which we, according to the usual definition in EPMA, model by the ratio of a measured $I_{(Z,j)}$ and a standard intensity $I^{\mathrm{std}}_{(Z,j)}$. Both intensities can be computed from the same model, only the underlying material description, $\rho(x)$ resp. $\rho^{\mathrm{std}}(x)$, changes.

A.1.1 Mass Concentrations

The mass concentration ρ_i of a constituent $i \in \{1, \dots, n_e\}$ in a compound with n_e elements is defined as the ratio of the mass M_i of the constituent i divided by the volume of the compound. If we consider mass concentration fields $\rho_i(x)$ at a given point $x \in \mathfrak{S} \subset \mathbb{R}^3$ with $\mathrm{d}x \subset \mathbb{R}^3$ a small volume around x, we consider mass and volume inside $\mathrm{d}x$

$$\rho_i(x) = \frac{M_i}{\mathrm{vol}(\mathrm{d}x)}. \tag{A.2}$$

The total mass of the compound is given by $M_{\mathrm{tot}} = \sum_{i=1}^{n_e} M_i$, hence the total density $\rho_{\mathrm{tot}}(x)$ of the compound is

$$\rho_{\mathrm{tot}}(x) = \frac{\sum_{i=1}^{n_e} M_i}{\mathrm{vol}(\mathrm{d}x)} = \sum_{i=1}^{n_e} \rho_i(x). \tag{A.3}$$

We introduce the notation $\rho_{\mathrm{tot}}(x)$ for the total density to distinguish it from the notation for the vector of mass concentrations $\rho(x) = (\rho_1(x), \dots \rho_{n_e}(x))^T$ which we use frequently.

In this work, we consider the mass concentrations as the general way to describe a material and derive other material quantities which are required for the forward model based on mass concentrations. This assumption is not generally applicable to all compounds (c.f. Thwaites [1983]), but it simplifies notation and all methods we describe in Chapter B can be modified to other material descriptions as well.

Modeling material quantities based on e.g. mass, volume or molar fractions is material specific, and should be implemented individually for a particular material; thereby considering e.g. variations in mass concentrations due to different atomic arrangements like lattices or molecules. We refer to Section A.3.1, where we present relations of mass concentrations to mass and volume fractions based on simple assumptions, which are applied in our simulations.

A.1.2 The Intensity of Generated X-Rays

The intensity of generated x-rays $I_{(Z,j)}$ is a composition of multiple consecutive model parts:

$$I_{(Z,j)} = \mathcal{A}_{(Z,j)}(\rho) \circ \mathcal{X}_{(Z,j)}(\rho) \circ \phi_{(Z,j)} \circ \Psi(\rho). \tag{A.4}$$

Beginning with the innermost part, they are:

- the transport of induced electrons inside the sample, quantified by the electron fluence $\Psi(\epsilon, x, \Omega) = |v(\epsilon)| n(\epsilon, x, \Omega)$. The electron fluence is given by the number density of electrons $n(\epsilon, x, \Omega)$ with energy ϵ at position x moving in direction Ω weighted with the velocity $v(\epsilon)$ of electrons;
- the ionization of atoms leading to the emission of x-rays, the ionization distribution $\phi(x)$;
- the combination of the ionization distribution $\phi(x)$ with the actual presence of atoms: the x-ray generation distribution $\mathcal{X}(x)$; and
- the attenuation $\mathcal{A}$ experienced by the x-rays due to absorption and scattering.

While in Equation (A.4) direct dependence of the model operators on the mass concentrations is highlighted by $\cdot(\rho)$, the list highlights the dependency of a realization of each operator on the electron energy ϵ, the position x and the direction Ω. Note that through the parts of the model, the dimensionality of the variables decreases from $(\epsilon, x, \Omega) \in \mathbb{R}^+ \times \mathbb{R}^3 \times S^2$ to a scalar $I_{(Z,j)} \in \mathbb{R}$. In the following the individual parts are discussed in further detail.

A.1.3 Mass Attenuation

X-rays, which are generated inside the sample, are attenuated while passing through to reach the detector. Attenuation is the combination of losses in x-ray intensity due to absorption and scattering. The attenuation of x-rays which are traveling along an axis (described by $z \in \mathbb{R}$) is commonly modeled using the Beer-Lambert law [Pinard, 2016]

$$\frac{\partial \mathcal{I}}{\partial z} = -\mu(z)\mathcal{I}(z), \tag{A.5}$$

a linear ODE, where $\mathcal{I}$ is the intensity and μ the linear attenuation coefficient. The ODE has the solution

$$\mathcal{I}(z) = \mathcal{I}_0 \exp(-\int_0^z \mu(\bar{z})\,\mathrm{d}\bar{z}), \tag{A.6}$$

which in the case of a constant attenuation coefficient μ simplifies to $\mathcal{I}(z) = \mathcal{I}_0 \exp(-\mu z)$. In our model, x-rays are generated from a continuous field, hence the initial intensity is $\mathcal{I}_0 = \mathcal{X}(x)$. The attenuation factor for a position x in the material is the line integral from x to the detector $\int_{d(x)} \cdot\,\mathrm{d}\bar{z}$. The detected intensity is then given by the integral over the whole material.

$$\mathcal{A}_{(Z,j)}(\mathcal{X}_{(Z,j)}) = \int_{\mathbb{R}^3} \exp\left(-\int_{d(x)} \mu_{(Z,j)}(\bar{z})\,\mathrm{d}\bar{z}\right) \mathcal{X}_{(Z,j)}(x)\,\mathrm{d}x. \tag{A.7}$$

The linear attenuation coefficient $\mu_{(Z,j)}(x)$ for compounds can be modeled by linear combination with mass concentrations ρ_i

$$\mu_{(Z,j)}(x) = \sum_{i=1}^{n_e} \rho_i(x) \left(\frac{\mu}{\rho}\right)_{i,(Z,j)}, \tag{A.8}$$

whereby the mass attenuation coefficients $\left(\frac{\mu}{\rho}\right)_{i,(Z,j)}$ for elements i are interpolated for the characteristic energy of an x-ray (Z, j). Implementations and data for mass absorption coefficients can be found in Ritchie [2021a,b]; Chantler et al. [2009].

A.1.4 The X-ray Generation Distribution

To compute the x-ray generation distribution for an x-ray (Z, j), we weigh the ionization distribution $\phi_{(Z,j)}$ by the number of atoms of element Z per unit volume $\frac{\rho_Z}{A_Z}$.

$$\mathcal{X}_{(Z,j)}(\phi_{(Z,j)}) = \frac{\rho_Z(x)}{A_Z}\phi_{(Z,j)}(x). \tag{A.9}$$

Thereby $\rho_Z(x)$ is the mass concentration and A_Z the atomic mass of element Z.

A.1.5 The Ionization Distribution

The ionization distribution field $\phi_{(Z,j)}(x)$ of x-ray (Z, j) is given by product of the number density of electrons $n(\epsilon, x, \Omega)$ traveling with velocity $v(\epsilon)$ in direction Ω and a cross-section $\sigma_{(Z,j)}(\epsilon)$ describing the fraction of collisions leading to the emission of x-rays.

$$\phi_{(Z,j)}(x) = \int_0^\infty \sigma_{(Z,j)}(\epsilon) \int_{S^2} \underbrace{|v(\epsilon)|n(\epsilon, x, \Omega)}_{=:\Psi(\epsilon,x,\Omega)} \,\mathrm{d}\Omega \,\mathrm{d}\epsilon \tag{A.10}$$

The cross-section $\sigma_{(Z,j)}(\epsilon)$ collects the ionization cross-section and the fluorescence yield. Implementations and data for both can be found in Ritchie [2021b, 2020]; Bote et al. [2009]; Bote and Salvat [2008]; Cullen et al. [1997].

Due to the high dimensionality ($\mathbb{R}^+ \times \mathbb{R}^3 \times S^2$) of the electron number density $n(\epsilon, x, \Omega)$ and the electron fluence $\Psi(\epsilon, x, \Omega)$, a simulation of either is complex. Hence, it is advantageous to derive a model for the averaged electron fluence. In Equation (A.10) the electron number density only appears weighted by the velocity $|v(\epsilon)|$ and averaged in direction $\Omega \in S^2$. We define

$$\psi_0^0(\epsilon, x) = \int_{S^2} |v(\epsilon)|n(\epsilon, x, \Omega) \,\mathrm{d}\Omega \tag{A.11}$$

and describe the P_N model, a moment expansion that governs ψ_0^0 (the zeroth moment of the electron fluence Ψ), in the next section.

A.1.6 The Electron Fluence: PN-Model

For a detailed derivation of the P_N model see Bünger et al. [2021]; Buenger et al. [2021]. Here only the expansion using the method of moments [Larsen et al., 1997] is described.

A.1.6.1 Continuous Slowing Down Approximation (CSD) of the Linear Boltzmann Equation

The Linear Boltzmann Equation [Cercignani, 1988] describes particle transport, where the self-interaction between particles can be neglected and only (elastic and inelastic) scattering with the background medium is considered. In the context of EPMA, the time dependence in the linear Boltzmann equation is negligible and only the stationary solution is sought-for. The fact that electrons most probably lose energy in a sequence of small energy losses,

justifies the replacement of the inelastic scattering operator by its continuous slowing down approximation (CSD). The continuous energy loss is modeled using the stopping power $S(\epsilon, x)$, the average energy loss per path length. The stopping power governs parts of the scattering operator in the linear Boltzmann equation, the leftover parts we call $\mathcal{Q}(\epsilon, x)[\Psi]$. The resulting transport equation is an evolution equation for the electron fluence $\Psi(\epsilon, x, \Omega) = |v(\epsilon)|n(\epsilon, x, \Omega)$ given by:

$$-\partial_\epsilon(S(\epsilon, x)\Psi(\epsilon, x, \Omega)) + \Omega\nabla_x\Psi(\epsilon, x, \Omega) = \mathcal{Q}(\epsilon, x)[\Psi(\epsilon, x, \Omega)]. \tag{A.12}$$

As mentioned, it is expensive to solve Equation (A.12) numerically, hence we will reduce the model complexity using a Galerkin method which expands $\Psi(\epsilon, x, \Omega)$ into a linear combination of moments and basis function in Ω. The expansion is called the method of moments [Larsen et al., 1997]. The spherical harmonics, an orthonormal set functions $S^2 \to \mathbb{R}$, have proven to be suitable basis functions, because they possess favorable properties which are inherited to the resulting model (the P_N model).

A.1.6.2 Digression: Spherical Harmonics

The real spherical harmonic [Müller, 1966]

$$Y_l^k : S^2 \to \mathbb{R} \tag{A.13}$$

of degree $l \in \mathbb{N}_0$ and of order k ($|k| \leq l$) maps a direction $\Omega(\mu, \varphi) \in S^2$, also represented by polar $\mu \in [0, \pi]$ and azimuthal (longitudinal) angle $\varphi \in [0, 2\pi]$, to the real numbers $\mathbb{R}$. The real spherical harmonic Y_l^k is given by

$$Y_l^k(\Omega(\mu, \varphi)) = C_l^{|k|} P_l^{|k|}(\cos(\mu)) \begin{cases} \cos(|k|\varphi) & k > 0 \\ \frac{1}{\sqrt{2}} & k = 0 \\ \sin(|k|\varphi) & k < 0 \end{cases}, \quad C_l^{|k|} = (-1)^{|k|}\sqrt{\frac{2l+1}{2\pi}\frac{(l-|k|)!}{(l+|k|)!}}, \tag{A.14}$$

where $C_l^{|k|}$ is a normalization constant and $P_l^{|k|}(\cos(\mu))$ is the associated Legendre polynomial. In Figure A.1 the spherical harmonics up to degree $l \leq 2$ are shown on the surface of a unit sphere.

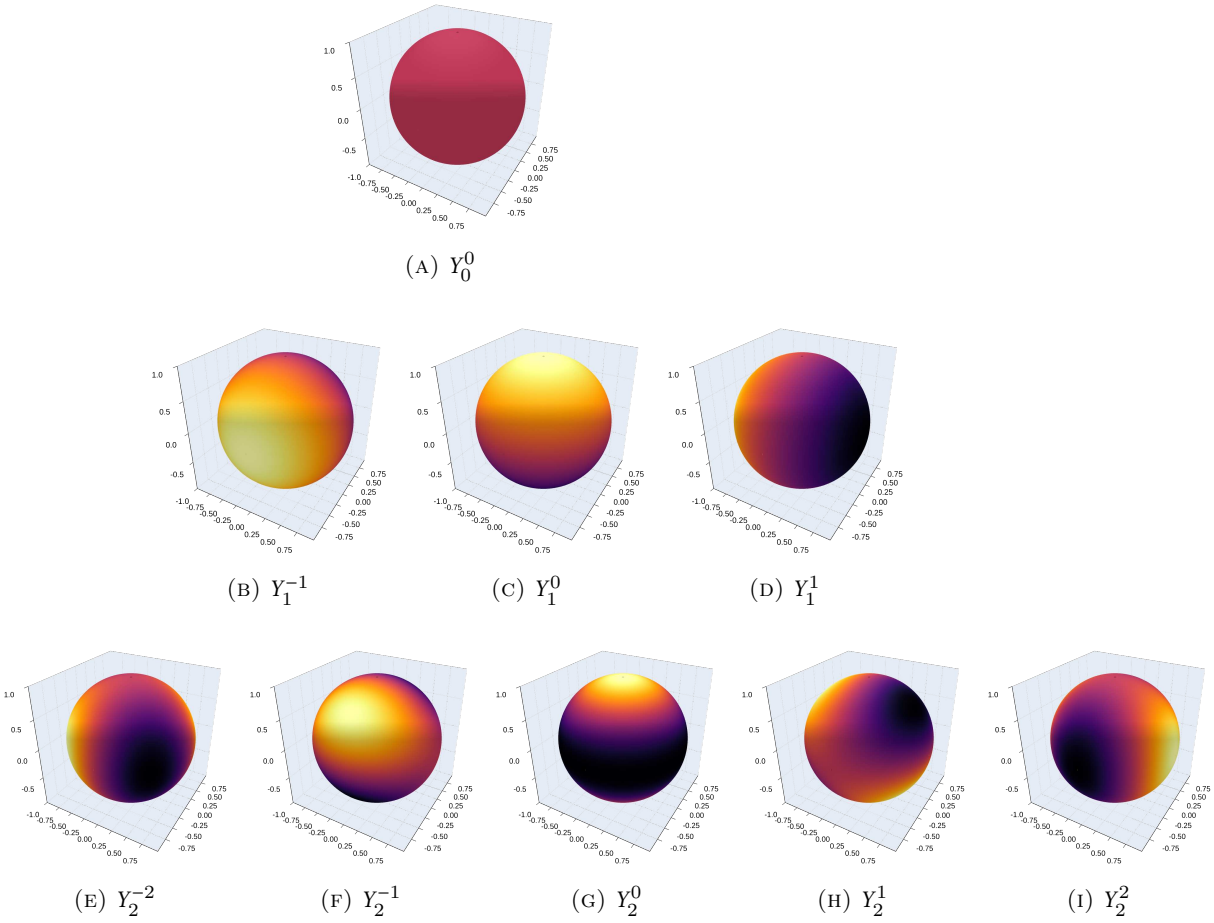
Properties of Spherical Harmonics We recapitulate some properties of the spherical harmonics, which are used to derive the P_N model.

- Spherical harmonics form an orthonormal set of basis functions over the unit sphere S^2

$$\int_{S^2} Y_l^k(\Omega) Y_{l'}^{k'}(\Omega)\, \mathrm{d}\Omega = \delta_{l,l'}\delta_{k,k'} = \begin{cases} 1 & l = l' \wedge k = k' \\ 0 & \text{else} \end{cases}. \tag{A.15}$$

- Spherical harmonics satisfy a recursion relation. The product of a spherical harmonic Y_l^k with a direction Ω can be expressed by the recursion

$$\Omega Y_l^k(\Omega) = \frac{1}{2}\begin{pmatrix} (1-\delta_{k,-1})(c_{l-1}^{|k|-1} Y_{l-1}^{k^-} - d_{l+1}^{|k|-1} Y_{l+1}^{k^-}) - e_{l-1}^{|k|+1} Y_{l-1}^{k^+} + f_{l+1}^{|k|+1} Y_{l+1}^{k^+} \\ \Theta(k)((1-\delta_{k,1})(-c_{l-1}^{|k|-1} Y_{l-1}^{-k^-} + d_{l+1}^{|k|-1} Y_{l+1}^{-k^-}) - e_{l-1}^{|k|+1} Y_{l-1}^{-k^+} + f_{l+1}^{|k|+1} Y_{l+1}^{-k^+}) \\ 2(a_{l-1}^k Y_{l-1}^k + b_{l+1}^k Y_{l+1}^k) \end{pmatrix}. \tag{A.16}$$



(A) Y_0^0

(B) Y_1^{-1} (C) Y_1^0 (D) Y_1^1

(E) Y_2^{-2} (F) Y_2^{-1} (G) Y_2^0 (H) Y_2^1 (I) Y_2^2

Figure A.1: *The spherical harmonics $Y_l^k(\Omega(\mu,\varphi))$ up to degree $l = 2$ visualized on a sphere. For each degree l, two additional orders k are added, hence the total number of spherical harmonics with $l \leq N$ is $(N+1)^2$.*

Ω	Y_l^k is even, if	Y_l^k is odd, if
e_x	$(k < 0 \wedge k \text{ odd}) \vee (k \geq 0 \wedge k \text{ even})$	$(k < 0 \wedge k \text{ even}) \vee (k \geq 0 \wedge k \text{ odd})$
e_y	$k \geq 0$	$k < 0$
e_z	$(l+k)$ even	$(l+k)$ odd

Table A.1: *Classification of the spherical harmonics in even (e) and odd (o) functions with respect to the Cartesian basis vectors (e_x, e_y and e_z).*

Thereby the right-hand side only depends on spherical harmonics of degree $l-1$ and $l+1$. The parameters used in this equation are:

$$\Theta(k) = \begin{cases} 1 & k \geq 0 \\ -1 & k < 0 \end{cases} \tag{A.17a}$$

$$k^+ = \begin{cases} k+1 & k \geq 0 \\ k-1 & k < 0 \end{cases} \tag{A.17b}$$

$$k^- = \begin{cases} k-1 & k \geq 0 \\ k+1 & k < 0 \end{cases} \tag{A.17c}$$

$$a_l^k = \sqrt{\frac{(l-k+1)(l+k+1)}{(2l+3)(2l+1)}} \tag{A.17d}$$

$$b_l^k = \sqrt{\frac{(l-k)(l+k)}{(2l+1)(2l-1)}} \tag{A.17e}$$

$$c_l^k = \sqrt{\frac{(l+k+1)(l+k+2)}{(2l+3)(2l+1)}} \begin{cases} 0 & k < 0 \\ \sqrt{2} & k = 0 \\ 1 & k > 0 \end{cases} \tag{A.17f}$$

$$d_l^k = \sqrt{\frac{(l-k)(l-k-1)}{(2l+1)(2l-1)}} \begin{cases} 0 & k < 0 \\ \sqrt{2} & k = 0 \\ 1 & k > 0 \end{cases} \tag{A.17g}$$

$$e_l^k = \sqrt{\frac{(l-k+1)(l-k+2)}{2l+1)(2l+1)}} \begin{cases} \sqrt{2} & k = 1 \\ 1 & k > 1 \end{cases} \tag{A.17h}$$

$$f_l^k = \sqrt{\frac{(l+k)(l+k-1)}{2l+1)(2l-1)}} \begin{cases} \sqrt{2} & k = 1 \\ 1 & k > 1 \end{cases} \tag{A.17i}$$

- All spherical harmonics are eigenfunctions of isotropic operators (isotropic in a sense, that they only depend on the angle $\Omega \cdot \Omega'$, not directly on the direction Ω).

$$\int_{S^2} \sigma(\Omega \cdot \Omega') Y_k^l(\Omega') \, \mathrm{d}\Omega' = \sigma_l Y_k^l(\Omega). \tag{A.18}$$

  The eigenvalue σ_l corresponding to the eigenfunction Y_l^k only depends on the degree l of the spherical harmonic.

- We augment the spherical harmonics with a classification in even (e) and odd (o) functions with respect to the Cartesian basis vectors (e_x, e_y and e_z). See Table A.1 for the distinction.

A.1.6.3 Spectral Galerkin Approximation

We apply the method of moments to the BCSD Equation (A.12). The electron fluence Ψ is approximated by a linear combination of spherical harmonics up to degree $l \leq N \in \mathbb{N}^0$

$$\Psi(\epsilon, x, \Omega) \approx \Psi_{P_N}(\epsilon, x, \Omega) = \sum_{l \leq N, |k| \leq l} \psi_l^k(\epsilon, x) Y_l^k(\Omega). \tag{A.19}$$

The coefficients $\psi_l^k(\epsilon, x)$ are the moments of the electron fluence Ψ and, since spherical harmonics form an orthonormal basis of S^2, the moment ψ_l^k is given by the scalar product

$$\psi_l^k(\epsilon, x) = \int_{S^2} \Psi(\epsilon, x, \Omega) Y_l^k(\Omega) \, \mathrm{d}\Omega. \tag{A.20}$$

Moments ψ_l^k of higher degree $l > N$ are assumed to be $\ll 1$, therefore negligible. The zeroth spherical harmonic $Y_0^0(\Omega)$ is constant, so we can identify $\psi_0^0(\epsilon, x)$ with the variable in Equation (A.10).

From Equation (A.12) evolution equations for the moments ψ_l^k are derived by testing (multiplying and integrating: $\int_{S^2} \cdot Y_{l'}^{k'}(\Omega) \, \mathrm{d}\Omega$) with spherical harmonics. Testing with the set of same spherical harmonics ($l' \leq N, |k'| \leq l'$) which are used in the approximation Ψ_{P_N} leads to a system of equations, the P_N model

$$-\partial_\epsilon(S\psi) + A^{(x)}\partial_x\psi + A^{(y)}\partial_y\psi + A^{(z)}\partial_z\psi + C\psi = 0. \tag{A.21}$$

The unknown variables are the coefficients $\psi = \{\psi_l^k\}_{l \in \mathbb{N}_0, |k| \leq l}$ used in the P_N approximation, the moments of the electron fluence Ψ. We derive the summands in Equation (A.21) from Equation (A.12) individually.

Stopping Power The first term in Equation (A.12) is linear in Ψ and the spherical harmonics form an orthonormal basis, hence

$$\begin{aligned}
&\int_{S^2} -\partial_\epsilon \left(S(\epsilon, x) \sum_{l \leq N, |k| \leq l} \psi_l^k(\epsilon, x) Y_l^k(\Omega) \right) Y_{l'}^{k'}(\Omega) \, \mathrm{d}\Omega \\
&\qquad = -\partial_\epsilon \left(S(\epsilon, x) \sum_{l \leq N, |k| \leq l} \psi_l^k(\epsilon, x) \underbrace{\int_{S^2} Y_l^k(\Omega) Y_{l'}^{k'}(\Omega) \, \mathrm{d}\Omega}_{\delta_{l,l'}\delta_{k,k'}} \right) \\
&\qquad = -\partial_\epsilon (S(\epsilon, x) \psi_{l'}^{k'}(\epsilon, x)).
\end{aligned} \tag{A.22}$$

The stopping power S is a material property, where for a compound we assume (c.f. Section A.1.1) additivity using the mass concentrations $\rho_i(x)$

$$S(\epsilon, x) = \sum_{i=1}^{n_e} \rho_i(x) S_i(\epsilon). \tag{A.23}$$

Thereby $S_i(\epsilon)$ is a specific stopping power of pure element i. For the stopping power of pure elements there exists multiple models, e.g. a model based on the Bethe-Loss formula [Reimer, 1998] or interpolation tables from a more advanced model [Cullen et al., 1997; Bünger, 2021]; the latter we use in our current implementation.

Transport Coefficient The transport coefficient C is derived from the scattering operator $\mathcal{Q}(\epsilon, x)[\Psi]$. In Buenger et al. [2021] the operator is given as

$$\mathcal{Q}(\epsilon, x)[\Psi(\epsilon, x)] = \int_{S^2} \sigma_s(\epsilon, x, \Omega' \cdot \Omega) \Psi(\epsilon, x, \Omega') \, \mathrm{d}\Omega' - \sigma_t(\epsilon, x) \Psi(\epsilon, x, \Omega), \tag{A.24}$$

where σ_s and σ_t are scattering cross-sections of elastic and inelastic collisions. The scattering operator $\mathcal{Q}$ is linear in the electron fluence Ψ, so if we replace Ψ with its P_N approximation

and test with the spherical harmonic $Y_{l'}^{k'}$, we derive the transport coefficient matrix C by similar argument as in Equation (A.22).

$$\begin{aligned} C_{(l,k),(l',k')} &= -\int_{S^2} Q(\epsilon,x)[Y_l^k]Y_{l'}^{k'} \\ &= -\int_{S^2}\left(\int_{S^2}\sigma_s(\epsilon,x,\Omega\cdot\Omega')Y_l^k(\Omega')\,\mathrm{d}\Omega' - \sigma_t(\epsilon,x)Y_l^k(\Omega)\right)Y_{l'}^{k'}\,\mathrm{d}\Omega \\ &= -(\sigma_l(\epsilon,x)-\sigma_t(\epsilon,x))\delta_{(l,k),(l',k')}. \end{aligned} \tag{A.25}$$

Due to the eigenfunction property of spherical harmonics, C is diagonal with entries $\sigma_l - \sigma_t$, which are the eigenvalues of the scattering operator Q corresponding only to the degree l of the spherical harmonic Y_l^k. As for the stopping power, we assume additivity for the transport coefficients using the mass concentrations $\rho_i(x)$

$$(\sigma_l-\sigma_t)(\epsilon,x) = \sum_{i=1}^{n_e}\rho_i(x)\underbrace{(\sigma_l-\sigma_t)_i(\epsilon)}_{:=C_{l,i}(\epsilon)}. \tag{A.26}$$

The specific coefficients $C_{l,i}(\epsilon) = (\sigma_l-\sigma_t)_i(\epsilon)$ for the pure elements are derived from scattering cross-sections generated using the code from Salvat et al. [2005], which in Bünger [2021] are tabulated for the different degrees of the spherical harmonics.

Advection Matrices By testing the advection operator $\Omega\cdot\nabla_x\Psi$ with the spherical harmonic $Y_{l'}^{k'}$, the advection matrices $A^{(x)}$, $A^{(y)}$ and $A^{(z)}$ are identified using the recursion relation given in Equation (A.16) and the coefficients in Equation (A.17).

$$(A^{(x)})_{(l,k),(l',k')} = \int_{S^2}\Omega_1 Y_l^k Y_{l'}^{k'}\,\mathrm{d}\Omega = \frac{1}{2}\begin{cases} \begin{cases} c_{l-1}^{|k|-1} & k' = k^- \wedge k \neq -1 \\ -e_{l-1}^{|k|+1} & k' = k^+ \end{cases} & l' = l-1 \\ 0 \quad \text{else} & \\ \begin{cases} -d_{l+1}^{|k|-1} & k = k^- \wedge k \neq -1 \\ f_{l+1}^{|k|+1} & k = k^+ \end{cases} & l' = l+1 \\ 0 \quad \text{else} & \\ 0 & \text{else} \end{cases} \tag{A.27a}$$

$$(A^{(y)})_{(l,k),(l',k')} = \int_{S^2}\Omega_2 Y_l^k Y_{l'}^{k'}\,\mathrm{d}\Omega = \frac{\Theta(k)}{2}\begin{cases} \begin{cases} -c_{l-1}^{|k|-1} & k' = -k^- \wedge k \neq 1 \\ -e_{l-1}^{|k|+1} & k' = -k^+ \end{cases} & l' = l-1 \\ 0 \quad \text{else} & \\ \begin{cases} d_{l+1}^{|k|-1} & k = -k^- \wedge k \neq 1 \\ f_{l+1}^{|k|+1} & k = -k^+ \end{cases} & l' = l+1 \\ 0 \quad \text{else} & \\ 0 & \text{else} \end{cases} \tag{A.27b}$$

$$(A^{(z)})_{(l,k),(l',k')} = \int_{S^2}\Omega_3 Y_l^k Y_{l'}^{k'}\,\mathrm{d}\Omega = \begin{cases} a_{l-1}^k & k' = k \wedge l' = l-1 \\ b_{l+1}^k & k' = k \wedge l' = l+1 \\ 0 & \text{else} \end{cases} \tag{A.27c}$$

Note that the advection matrices are very sparse and possess a specific pattern with respect to the classification of the spherical harmonics (Table A.1). The advection matrix in a direction $d \in \{e_x, e_y, e_z\}$ does only have non-zero entries if the e/o-classifications of the respective spherical harmonic functions Y_l^k and $Y_{l'}^{k'}$ differ in one direction d and coincide in the other two Cartesian directions. Furthermore, all advection matrices are symmetric $A^{(n)T} = A^{(n)}$, which can quickly be verified by their definition using $Y_l^k Y_{l'}^{k'} = Y_{l'}^{k'} Y_l^k$. This sparsity of the advection matrices is exploited in the numerical method and paves the way towards an efficient implementation of the P_N model.

A.1.7 Beam Modeling: PN-Boundary Conditions

Initial conditions for the electron fluence Ψ in energy are specified at a given maximum energy ϵ_{init}.

$$\Psi(\epsilon_{\text{init}}, x, \Omega) = \Psi_0(x, \Omega). \tag{A.28}$$

The translation of an initial condition in energy $\Psi_0(x, \Omega)$ into its moments $\psi_{l,0}^k(x)$ is straightforward. Using Equation (A.20), the initial condition Ψ_0 is expanded into the respective moments. In the following experiments, we will typically use $\Psi_0(x, \Omega) = 0$, assuming that no beam electrons are present inside the sample at $\epsilon = \epsilon_{\text{init}}$.

For an implementation, the spatial variable x is confided to $\mathfrak{S} \subset \mathbb{R}^3$. Then a proper treatment of boundary conditions in space is necessary. At a point $x \in \partial\mathfrak{S}$ with n the outward boundary normal, we are allowed to prescribe the ingoing part of the electron fluence Ψ

$$\Psi(\epsilon, x, \Omega) = \Psi_{\text{in}}(\epsilon, x, \Omega) \quad \Omega \cdot n < 0. \tag{A.29}$$

In Bünger et al. [2021] boundary conditions are derived by testing Equation (A.29) with the spherical harmonics of odd (o) classification in the respective direction of the boundary normal n. The boundary conditions for the moments ψ are given by

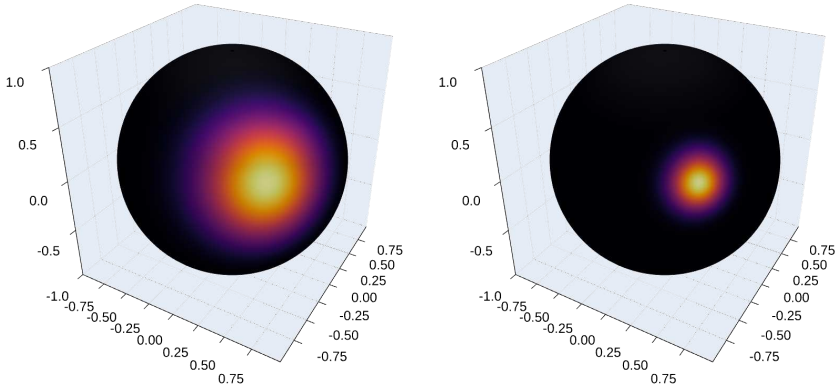
$$\psi_{o,n}(\epsilon, x) = L_o^{(n)} A_{o,e}^{(n)} \psi_{e,n}(\epsilon, x) + g_{o,n}(\epsilon, x) \quad x \in \partial\mathfrak{S}, \tag{A.30}$$

where $\psi_{o,n}$ are the moments corresponding to odd spherical harmonics and $\psi_{e,n}$ the moments corresponding to even spherical harmonics in the direction n. The matrix $A_{o,e}^{(n)}$ is the block of the advection matrices which maps even (in direction n) moments to odd moments (in direction n). Direct formulas for $L_o^{(n)}$ are taken from Bünger [2021] and reproduced in Section C.1. The source $g_{o,n}$ is derived by testing the ingoing electron fluence Ψ_{in} with spherical harmonics $Y_{o,n}$ which are odd in the direction of the boundary normal n

$$g_{o,n}(\epsilon, x) = \int_{n\cdot\Omega<0} Y_{o,n}(\Omega) \Psi_{\text{in}}(\epsilon, x, \Omega) \, \mathrm{d}\Omega. \tag{A.31}$$

Beam Parameters The electron beam hits the sample at a surface (in our examples usually $\{(x_1, x_2, x_3) \in \mathfrak{S} | x_1 = 0\}$). We model the beam using an isotropic Gaussian distribution on the material surface, a Gaussian distribution in energy and a three-dimensional Von-Mises-Fisher distribution (a distribution defined on S^2) in direction. Using respective means μ_x and μ_ϵ, variances σ_x and σ_ϵ as well as the mean direction μ_Ω and concentration parameter κ of the Von-Mises-Fisher distribution, we define

$$\Psi_{\text{in}}(\epsilon, x, \Omega) = \mathcal{N}\left((x_2, x_3)^T | \mu_x, \mathrm{diag}(\sigma_x)\right) \mathcal{N}(\epsilon | \mu_\epsilon, \sigma_\epsilon) \mathcal{F}(\Omega | \mu_\Omega, \kappa), \tag{A.32}$$



(A) $\kappa = 10$. (B) $\kappa = 50$

Figure A.2: *The probability density function of the Von-Mises-Fisher distribution visualized on a sphere for different values of the concentration parameter κ. It illustrates the dependence of the incoming beam electron fluence Ψ_{in} on the direction Ω.*

where $\mathcal{N}$ is the probability density function of the Gaussian distribution and $\mathcal{F}$ is the probability density function of the Von-Mises-Fisher distribution. Integration of $\mathcal{F}$ into the respective moments $g^{(n)}(\epsilon, x)$ is performed numerically.

A.2 Numerical Methods for the K-Ratio Model

This section covers the numerical methods to compute k-ratio using the model presented in Section A.1. In Section A.2.1 we describe StaRMAP, a numerical method for the solution of the moment expansion for the electron fluence, based on a generic moment equation and detail on the solution of the P_N-equation in Section A.2.2. In Sections A.2.3 and A.2.4 we describe the numerical methods for the ionization distribution and attenuation.

A.2.1 StaRMAP: A second order Staggered Grid Method for Radiation Moment Approximation

This section describes the numerical method StaRMAP, which solves spherical harmonic moment systems. The main characteristic of StaRMAP is the discretization of the solution variable on staggered grids. Allowed by a specific structure of the equation, the staggered grid discretization leads to an efficient second order approximation. It is originally published in Seibold and Frank [2014] and already used in the context of electron transport in EPMA in Bünger [2021]; Bünger et al. [2021]; Buenger et al. [2021]. We describe StaRMAP based on the generic moment equation:

$$\begin{aligned} S\partial_t\psi + A^{(x)}\partial_x\psi + A^{(y)}\partial_y\psi + A^{(z)}\partial_z\psi + C\psi = Q \quad \forall t \in \mathbb{R}, x \in \mathfrak{S} \subset \mathbb{R}^3 \\ \psi_{o,n}(t,x) = M^{(n)}_{o,e}\psi_{e,n}(t,x) + g_{o,n}(t,x) \quad \forall t \in \mathbb{R}, x \in \partial\mathfrak{S}. \end{aligned} \tag{A.33}$$

Later we will show how to adapt the P_N-equation (Equation (A.21)) to fit this generic equation.

The solution variable $\psi(t,x) : \mathbb{R} \times \mathbb{R}^3 \to \mathbb{R}^{\mathcal{N}}$ is subdivided into variables described by disjoint index sets $\{eee, eoo, oeo, ooe, oee, eoe, eeo, ooo\}$. The notation is interpreted as a classification in even (e) and odd (o) variables with respect to a space dimension, so that e.g. eoo is the index set being even in direction x, odd in direction y and odd in direction z. We assume an ordering of the solution variable ψ

$$\psi(t,x) = \begin{pmatrix} \psi_{eee}(t,x) \in \mathbb{R}^{\mathcal{N}_{eee}} \\ \psi_{eoo}(t,x) \in \mathbb{R}^{\mathcal{N}_{eoo}} \\ \vdots \\ \psi_{ooo}(t,x) \in \mathbb{R}^{\mathcal{N}_{ooo}} \end{pmatrix} : \mathbb{R} \times \mathbb{R}^3 \to \mathbb{R}^{\mathcal{N}}. \tag{A.34}$$

With respect to the ordering, the matrices in Equation (A.33) are assumed to possess specific patterns: The matrices $S(t,x) : \mathbb{R} \times \mathbb{R}^3 \to \mathbb{R}^{\mathcal{N}\times\mathcal{N}}$ and $C(t,x) : \mathbb{R} \times \mathbb{R}^3 \to \mathbb{R}^{\mathcal{N}\times\mathcal{N}}$ are diagonal; the source vector $Q(t,x) : \mathbb{R} \times \mathbb{R}^3 \to \mathbb{R}^{\mathcal{N}}$ is arbitrary; and the advection matrices $A^{(x)}$, $A^{(y)}$ and $A^{(z)} \in \mathbb{R}^{\mathcal{N}\times\mathcal{N}}$ are assumed to be of block sparse form, with respect to the e/o index sets. The only blocks, which are allowed to have non-zero entries, are blocks with differing e/o classification in the direction of the advection matrix (x, y or z) and coinciding $e/o/$ classification in the other two directions. For example the advection matrix in x direction $A^{(x)}$ can only possess non-zero values in the block $A^{(x)}_{e\star\bullet,o\star\bullet}$ and in the block $A^{(x)}_{o\star\bullet,e\star\bullet}$, where $\star, \bullet \in \{e, o\}$. All other parts of the advection matrix $A^{(x)}_{e\cdot\cdot,e\cdot\cdot}$ and $A^{(x)}_{o\cdot\cdot,o\cdot\cdot}$ are 0.

Regarding the ordering of the solution variable Equation (A.34), we can exemplarily write

$$A^{(x)} = \begin{pmatrix} 0 & 0 & 0 & 0 & A^{(x)}_{eee,oee} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & A^{(x)}_{eoo,ooo} \\ 0 & 0 & 0 & 0 & 0 & 0 & A^{(x)}_{oeo,eeo} & 0 \\ 0 & 0 & 0 & 0 & 0 & A^{(x)}_{ooe,eoe} & 0 & 0 \\ A^{(x)}_{oee,eee} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & A^{(x)}_{eoe,ooe} & 0 & 0 & 0 & 0 \\ 0 & 0 & A^{(x)}_{eeo,oeo} & 0 & 0 & 0 & 0 & 0 \\ 0 & A^{(x)}_{ooo,eoo} & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}. \tag{A.35}$$

The advection matrices in y and z direction can be written similarly.

A.2.1.1 Grid Structure

Consider the domain as a cuboid $\mathfrak{S} = (x_L, x_U) \times (y_L, y_U) \times (z_L, z_U) \in \mathbb{R}^3$. Per dimension $d \in \{x, y, z\}$ we introduce an even and an odd linearly spaced grid. The endpoints of the even grid coincide with the limits of the domain $\mathfrak{S}$, the points of the odd grid are shifted by a half-step $\frac{\Delta d}{2}$. We define the grid points as

$$g^{(d)}_e = \{(d_L + (i-1)\Delta d) | i \in \{1, \dots, n_d\}\}, \quad d \in \{x, y, z\} \tag{A.36a}$$

$$g^{(d)}_o = \{(d_L + (i - \frac{3}{2})\Delta d) | i \in \{1, \dots, n_d + 1\}\}, \quad d \in \{x, y, z\}. \tag{A.36b}$$

The step size for both grids is $\Delta d = \frac{d_U - d_L}{n_d - 1}, \quad d \in \{x, y, z\}$. From the scalar grids $g^{(d)}_{e/o}$ we generate eight 3D grids $G_{...}$ using the Cartesian product

$$\begin{aligned} G_{eee} &= g^{(x)}_e \times g^{(y)}_e \times g^{(z)}_e \\ G_{eoo} &= g^{(x)}_e \times g^{(y)}_o \times g^{(z)}_o \\ &\vdots \\ G_{ooo} &= g^{(x)}_o \times g^{(y)}_o \times g^{(z)}_o. \end{aligned} \tag{A.37}$$

In Figure A.3 the grid coordinates and the domain boundary are visualized, with the distinction in even and odd grids. Because of ghost points (outside the domain boundaries) which will be used for boundary conditions, odd grids contain more points than the respective even grids $|g^{(d)}_o| = n_d + 1 = |g^{(d)}_e| + 1$.

We discretize the components of the solution variable $\psi_{...}$ on the corresponding grids $G_{...}$, thus at a given point on the Grid $G_{\bullet\star\triangleright}$ ($\star, \bullet, \triangleright \in \{e, o\}$) only the discretized values of the subset $\psi_{\bullet\star\triangleright}$ of the solution variable are explicitly known. We denote indexing of the solution variable by e.g. $(\psi_{eeo})_{i,j,k}$ where $i \in \{1, \dots, n_x\}$, $j \in \{1, \dots, n_y\}$ and $k \in \{1, \dots, n_z + 1\}$.

A.2.1.2 Finite Difference Space Derivatives

Because of the structure of the advection matrices $A^{(d)}$ the time derivative of the solution variable e.g. $\partial_t \psi_{eee}$ only depends on spatial derivatives of solution variables of the switched e/o classification, e.g. $\partial_x \psi_{oee}$. The discretization on staggered grids allows us to define half-step central finite difference operators, which approximate the derivative at points located between the discretization points of e.g. ψ_{oee}, exactly where $\partial_t \psi_{eee}$ is discretized. Hence,

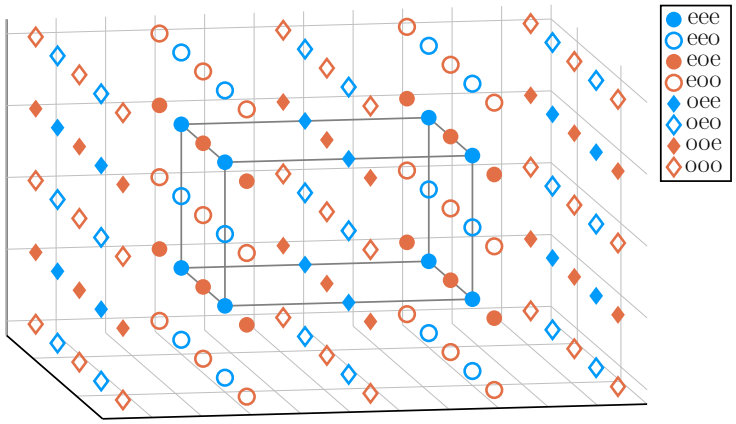


Figure A.3: *The spatial coordinates of the 3D staggered grids. The e/o property in the respective dimensions is illustrated by properties(color, shape and fill/stroke) of the marker(x-direction: e circles, o diamonds; y-direction: e blue, o red; z-direction: e filled, o stroked) For example the filled blue circles are the grid points of G_{eee}, which reside on the vertices of the domain (black box). The total number of grid points is the minimal number possible to discretize a 3D cuboid, hence the eee grid only discretizes the vertices of the domain. Grid points outside the domain are considered ghost-points, nevertheless they are equally structured to the grid points inside the domain.*

we can define a semi-discretization of Equation (A.33) (only the spatial derivatives are discretized) by

$$\begin{aligned}
&S_{eee}\partial_t\psi_{eee} + A^{(x)}_{eee,oee}D^{(x)}_{o\to e}\psi_{oee} + A^{(y)}_{eee,eoe}D^{(y)}_{o\to e}\psi_{eoe} + A^{(z)}_{eee,eeo}D^{(z)}_{o\to e}\psi_{eeo} + C_{eee}\psi_{eee} = Q_{eee}\\
&S_{eoo}\partial_t\psi_{eoo} + A^{(x)}_{eoo,ooo}D^{(x)}_{o\to e}\psi_{ooo} + A^{(y)}_{eoo,eeo}D^{(y)}_{e\to o}\psi_{eeo} + A^{(z)}_{eoo,eoe}D^{(z)}_{e\to o}\psi_{eoe} + C_{eoo}\psi_{eoo} = Q_{eoo}\\
&\qquad\vdots\\
&S_{ooo}\partial_t\psi_{ooo} + A^{(x)}_{ooo,eoo}D^{(x)}_{e\to o}\psi_{eoo} + A^{(y)}_{ooo,oeo}D^{(y)}_{e\to o}\psi_{oeo} + A^{(z)}_{ooo,ooe}D^{(z)}_{e\to o}\psi_{ooe} + C_{ooo}\psi_{ooo} = Q_{ooo}
\end{aligned} \tag{A.38}$$

where the finite difference operator $D^{(d)}_{\star\to\bullet}$ maps from a discretization on $\star \in \{e,o\}$ to a discretization on $\bullet \in \{o,e\}$ grids in the respective dimension d of the operator. We define then as (exemplarily in x-direction, y and z follow similarly)

$$\left(D^{(x)}_{e\to o}\psi_{eee}\right)_{i,j,k} = \begin{cases} \frac{1}{\Delta x}\left((\psi_{eee})_{i,j,k} - (\psi_{eee})_{i-1,j,k}\right) & i \in \{2,\dots,n_x\} \\ 0 & i \in \{1, n_x+1\} \end{cases} \tag{A.39a}$$

$$\left(D^{(x)}_{o\to e}\psi_{oee}\right)_{i,j,k} = \frac{1}{\Delta x}\left((\psi_{oee})_{i+1,j,k} - (\psi_{oee})_{i,j,k}\right) \quad i \in \{1,\dots,n_x\} \tag{A.39b}$$

For rigorous definition, derivatives at ghost points are defined to be zero, c.f. Equation (A.39a). However, the spatial derivatives at ghost points are not required because the value of ghost points is determined by the boundary conditions.

A.2.1.3 Time Discretization / Stepping Scheme

We describe the evolution of the solution variable ψ for one time step $[t, t+\Delta t]$. For the convenient notation of the time discretization, we define new index sets $\mathfrak{e} = eee \cup eoo \cup oeo \cup ooe$

and $\mathbb{o} = oee \cup eoe \cup eeo \cup ooo$ based on the e/o classifications

$$\psi = \begin{pmatrix} \psi_{\mathbb{e}} \\ \psi_{\mathbb{o}} \end{pmatrix} \quad \psi_{\mathbb{e}} = \begin{pmatrix} \psi_{eee} \\ \psi_{eoo} \\ \psi_{oeo} \\ \psi_{ooe} \end{pmatrix} \quad \psi_{\mathbb{o}} = \begin{pmatrix} \psi_{oee} \\ \psi_{eoe} \\ \psi_{eeo} \\ \psi_{ooo} \end{pmatrix}. \tag{A.40}$$

The solution components in $\psi_{\mathbb{e}}$ and $\psi_{\mathbb{o}}$ are not discretized on the same grids, nevertheless we collect them for convenient notation. The advection matrices $A^{(d)}$ are still of block-sparse form with respect to the index sets $\mathbb{e}$ and $\mathbb{o}$ in the sense, that $A^{(d)}_{\star,\star} = 0$ for $\star = \{\mathbb{e}, \mathbb{o}\}$ and $d = \{x, y, z\}$. Hence, we can write the semi-discretization (Equation (A.38)) as

$$\begin{aligned} \begin{pmatrix} S_{\mathbb{e}} & 0 \\ 0 & S_{\mathbb{o}} \end{pmatrix} \partial_t \begin{pmatrix} \psi_{\mathbb{e}} \\ \psi_{\mathbb{o}} \end{pmatrix} &+ \begin{pmatrix} 0 & A^{(x)}_{\mathbb{e},\mathbb{o}} \\ A^{(x)}_{\mathbb{o},\mathbb{e}} & 0 \end{pmatrix} \begin{pmatrix} D^{(x)}_{\mathbb{e}\to\mathbb{o}} & 0 \\ 0 & D^{(x)}_{\mathbb{o}\to\mathbb{e}} \end{pmatrix} \begin{pmatrix} \psi_{\mathbb{e}} \\ \psi_{\mathbb{o}} \end{pmatrix} \\ &+ \begin{pmatrix} 0 & A^{(y)}_{\mathbb{e},\mathbb{o}} \\ A^{(y)}_{\mathbb{o},\mathbb{e}} & 0 \end{pmatrix} \begin{pmatrix} D^{(y)}_{\mathbb{e}\to\mathbb{o}} & 0 \\ 0 & D^{(y)}_{\mathbb{o}\to\mathbb{e}} \end{pmatrix} \begin{pmatrix} \psi_{\mathbb{e}} \\ \psi_{\mathbb{o}} \end{pmatrix} \\ &+ \begin{pmatrix} 0 & A^{(z)}_{\mathbb{e},\mathbb{o}} \\ A^{(z)}_{\mathbb{o},\mathbb{e}} & 0 \end{pmatrix} \begin{pmatrix} D^{(z)}_{\mathbb{e}\to\mathbb{o}} & 0 \\ 0 & D^{(z)}_{\mathbb{o}\to\mathbb{e}} \end{pmatrix} \begin{pmatrix} \psi_{\mathbb{e}} \\ \psi_{\mathbb{o}} \end{pmatrix} \\ &+ \begin{pmatrix} C_{\mathbb{e}} & 0 \\ 0 & C_{\mathbb{o}} \end{pmatrix} \begin{pmatrix} \psi_{\mathbb{e}} \\ \psi_{\mathbb{o}} \end{pmatrix} = \begin{pmatrix} Q_{\mathbb{e}} \\ Q_{\mathbb{o}} \end{pmatrix}. \end{aligned} \tag{A.41}$$

Where we exemplarily denote the advection matrix (c.f. with Equation (A.35))

$$A^{(x)}_{\mathbb{e},\mathbb{o}} = \begin{pmatrix} A^{(x)}_{eee,oee} & 0 & 0 & 0 \\ 0 & 0 & 0 & A^{(x)}_{eoo,ooo} \\ 0 & 0 & A^{(x)}_{oeo,eeo} & 0 \\ 0 & A^{(x)}_{ooe,eoe} & 0 & 0 \end{pmatrix} \tag{A.42}$$

and the finite difference operators

$$D^{(x)}_{\mathbb{e}\to\mathbb{o}} = \mathrm{diag}(D^{(x)}_{e\to o}, D^{(x)}_{e\to o}, D^{(x)}_{o\to e}, D^{(x)}_{o\to e}) \tag{A.43a}$$

$$D^{(x)}_{\mathbb{o}\to\mathbb{e}} = \mathrm{diag}(D^{(x)}_{o\to e}, D^{(x)}_{e\to o}, D^{(x)}_{e\to o}, D^{(x)}_{o\to e}). \tag{A.43b}$$

In the time interval $[t, t + \Delta t]$ the equation components $S_{\mathbb{e}}(t,x)$, $S_{\mathbb{o}}(t,x)$, $C_{\mathbb{e}}(t,x)$, $C_{\mathbb{o}}(t,x)$, $Q_{\mathbb{e}}(t,x)$ and $Q_{\mathbb{o}}(t,x)$ are approximated constant and associated with $t + \frac{\Delta t}{2}$.

In Equation (A.41) the $\mathbb{e}$ part and the $\mathbb{o}$ part of the solution variable couple only due to the $\partial_t \psi$ term. If either $\psi_{\mathbb{e}}$ or $\psi_{\mathbb{o}}$ is assumed to be frozen $\partial_t \cdot = 0$, the equations decouple into two ordinary differential equation (ODE) systems.

$$S_{\mathbb{e}} \partial_t \psi_{\mathbb{e}} + C_{\mathbb{e}} \psi_{\mathbb{e}} = Q_{\mathbb{e}} - A^{(x)}_{\mathbb{e},\mathbb{o}} D^{(x)}_{\mathbb{o}\to\mathbb{e}} \psi_{\mathbb{o}} - A^{(y)}_{\mathbb{e},\mathbb{o}} D^{(y)}_{\mathbb{o}\to\mathbb{e}} \psi_{\mathbb{o}} - A^{(z)}_{\mathbb{e},\mathbb{o}} D^{(z)}_{\mathbb{o}\to\mathbb{e}} \psi_{\mathbb{o}} \tag{A.44a}$$

$$S_{\mathbb{o}} \partial_t \psi_{\mathbb{o}} + C_{\mathbb{o}} \psi_{\mathbb{o}} = Q_{\mathbb{o}} - A^{(x)}_{\mathbb{o},\mathbb{e}} D^{(x)}_{\mathbb{e}\to\mathbb{o}} \psi_{\mathbb{e}} - A^{(y)}_{\mathbb{o},\mathbb{e}} D^{(y)}_{\mathbb{e}\to\mathbb{o}} \psi_{\mathbb{e}} - A^{(z)}_{\mathbb{o},\mathbb{e}} D^{(z)}_{\mathbb{e}\to\mathbb{o}} \psi_{\mathbb{e}} \tag{A.44b}$$

Since for each equation the part of the solution vector which appears on the right-hand side of Equation (A.44a) or Equation (A.44b) is frozen, both ODEs are of very simple structure. Additional to the constant right-hand side, the matrices $S_{\mathbb{e}}$, $S_{\mathbb{o}}$, $C_{\mathbb{e}}$ and $C_{\mathbb{o}}$ are diagonal, then each of Equation (A.44a) and Equation (A.44b) are a set of scalar equations of the form

$$s \partial_t f(t) + c f(t) = r. \tag{A.45}$$

Assuming that $s \neq 0$ this equation has the analytical solution

$$f(t) = k \exp(\frac{-ct}{s}) + \frac{r}{c}, \tag{A.46}$$

where $k \in \mathbb{R}$. From a given $f(t)$ at time t, we can compute $f(t + \Delta t)$ as

$$\begin{aligned} f(t+\Delta t) &= k \exp(\frac{-c(t+\Delta t)}{s}) + \frac{r}{c} \\ &= (f(t) - \frac{r}{c}) \exp(\frac{-c\Delta t}{s}) + \frac{r}{c} \\ &= f(t) + (f(t) - \frac{r}{c}) \exp(\frac{-c\Delta t}{s}) - (f(t) - \frac{r}{c}) \\ &= f(t) + \frac{\Delta t}{s}(r - cf(t))E(\frac{-c\Delta t}{s}) \end{aligned} \tag{A.47}$$

where $E(c) = \frac{\exp(c)-1}{c}$. Using Equation (A.47) we can define update operators for the two ODE systems Equation (A.44a) and Equation (A.44b)

$$U_{\mathbb{e}}^{\Delta t}\begin{pmatrix}\psi_{\mathbb{e}}\\ \psi_{\mathbb{o}}\end{pmatrix} = \begin{pmatrix}\psi_{\mathbb{e}}\\ \psi_{\mathbb{o}}\end{pmatrix} + \Delta t \begin{pmatrix} S_{\mathbb{e}}^{-1}(R_{\mathbb{e}} - C_{\mathbb{e}}\psi_{\mathbb{e}})E(-C_{\mathbb{e}}S_{\mathbb{e}}^{-1}\Delta t) \\ 0 \end{pmatrix} \tag{A.48a}$$

$$U_{\mathbb{o}}^{\Delta t}\begin{pmatrix}\psi_{\mathbb{e}}\\ \psi_{\mathbb{o}}\end{pmatrix} = \begin{pmatrix}\psi_{\mathbb{e}}\\ \psi_{\mathbb{o}}\end{pmatrix} + \Delta t \begin{pmatrix} 0 \\ S_{\mathbb{o}}^{-1}(R_{\mathbb{o}} - C_{\mathbb{o}}\psi_{\mathbb{o}})E(-C_{\mathbb{o}}S_{\mathbb{o}}^{-1}\Delta t) \end{pmatrix} \tag{A.48b}$$

where $E(c)$ is applied elementwise to the diagonal matrices and the right-hand sides are denoted by

$$R_{\mathbb{e}} = Q_{\mathbb{e}} - A_{\mathbb{e},\mathbb{o}}^{(x)} D_{\mathbb{o}\to\mathbb{e}}^{(x)}\psi_{\mathbb{o}} - A_{\mathbb{e},\mathbb{o}}^{(y)} D_{\mathbb{o}\to\mathbb{e}}^{(y)}\psi_{\mathbb{o}} - A_{\mathbb{e},\mathbb{o}}^{(z)} D_{\mathbb{o}\to\mathbb{e}}^{(z)}\psi_{\mathbb{o}} \tag{A.49a}$$

$$R_{\mathbb{o}} = Q_{\mathbb{o}} - A_{\mathbb{o},\mathbb{e}}^{(x)} D_{\mathbb{e}\to\mathbb{o}}^{(x)}\psi_{\mathbb{e}} - A_{\mathbb{o},\mathbb{e}}^{(y)} D_{\mathbb{e}\to\mathbb{o}}^{(y)}\psi_{\mathbb{e}} - A_{\mathbb{o},\mathbb{e}}^{(z)} D_{\mathbb{e}\to\mathbb{o}}^{(z)}\psi_{\mathbb{e}}. \tag{A.49b}$$

To evolve a full time step $[t, t + \Delta t]$ for both parts of the solution $\psi_{\mathbb{e}}$ and $\psi_{\mathbb{o}}$, we apply Strang splitting and chain the update operators

$$\begin{pmatrix}\psi_{\mathbb{e}}(t+\Delta t)\\ \psi_{\mathbb{o}}(t+\Delta t)\end{pmatrix} = U^{\Delta t}\begin{pmatrix}\psi_{\mathbb{e}}(t)\\ \psi_{\mathbb{o}}(t)\end{pmatrix} = U_{\mathbb{o}}^{\frac{\Delta t}{2}} \circ U_{\mathbb{e}}^{\Delta t} \circ U_{\mathbb{o}}^{\frac{\Delta t}{2}} \begin{pmatrix}\psi_{\mathbb{e}}(t)\\ \psi_{\mathbb{o}}(t)\end{pmatrix}. \tag{A.50}$$

Due to the half-step central finite difference operators and the Strang splitting in time the resulting method is second order accurate [Bünger, 2021].

A.2.1.4 Boundary Conditions

Boundary values of even variables (even in the direction of the respective boundary) can be updated just as internal values, because odd variables are also defined on the ghost points and the spatial derivative can be approximated. However, for boundary values of odd variables special treatment is required. We prescribe values of odd variables on ghost points such that the boundary conditions from Equation (A.33)

$$\psi_{o,n}(t,x) = M_{o,e}^{(n)}\psi_{e,n}(t,x) + g_{o,n}(t,x) \tag{A.51}$$

are satisfied for interpolated odd variables. We derive them exemplary for odd variable values on ghost points in x-direction ($\star, \bullet \in \{e, o\}$).

$$(\psi_{o\star\bullet})_{x=x_L,j,k} \approx \frac{(\psi_{o\star\bullet})_{1,j,k} + (\psi_{o\star\bullet})_{2,j,k}}{2} = -M_{o\star\bullet,e\star\bullet}^{(x)}(\psi_{e\star\bullet})_{1,j,k} + g_{o\star\bullet,L}^{(x)} \tag{A.52a}$$

$$(\psi_{o\star\bullet})_{x=x_R,j,k} \approx \frac{(\psi_{o\star\bullet})_{n_x,j,k} + (\psi_{o\star\bullet})_{n_x+1,j,k}}{2} = M^{(x)}_{o\star\bullet,e\star\bullet}(\psi_{e\star\bullet})_{n_x,j,k} + g^{(x)}_{o\star\bullet,R} \tag{A.52b}$$

Thereby $M^{(n)}_{\dots,\dots}$ is the respective block of $M^{(n)}_{o,e}$. We can rewrite Equation (A.52) into explicit formulas for the boundary values $(\psi_{o\star\bullet})_{1,j,k}$ and $(\psi_{o\star\bullet})_{n_x+1,j,k}$.

$$(\psi_{o\star\bullet})_{1,j,k} = 2\left(-M^{(x)}_{o\star\bullet,e\star\bullet}(\psi_{e\star\bullet})_{1,j,k} + g^{(x)}_{o\star\bullet,L}\right) - (\psi_{o\star\bullet})_{2,j,k} \tag{A.53a}$$

$$(\psi_{o\star\bullet})_{n_x+1,j,k} = 2\left(M^{(x)}_{o\star\bullet,e\star\bullet}(\psi_{e\star\bullet})_{n_x,j,k} + g^{(x)}_{o\star\bullet,L}\right) - (\psi_{o\star\bullet})_{n_x,j,k} \tag{A.53b}$$

Similarly the boundary values in y and z-direction are derived.

A.2.1.5 Treatment of expm1div

In Equation (A.47) the function

$$E(c) = \frac{\exp(c) - 1}{c} \tag{A.54}$$

is introduced. For $c \to 0$ the fraction $E(c)$ goes to 1, which can be verified e.g. using the rule of L'Hospital. To avoid numerical instabilities due to $\frac{0}{0}$, we approximate $E(c)$ for small values of $c \ll 1$ by its Taylor expansion

$$E(c) = 1 + \frac{x}{2} + \frac{x^2}{6} + \frac{x^3}{24} + O(x^4). \tag{A.55}$$

A.2.2 Computing the Electron Fluence using StaRMAP

In order to solve the P_N-model using the method developed in Section A.2.1, we apply the product rule to the first term of Equation (A.21)

$$-\partial_\epsilon(S\psi) = -\partial_\epsilon S\psi - S\partial_\epsilon\psi. \tag{A.56}$$

The former term is combined with the transport coefficient C to form $C = -\partial_\epsilon S + C$. Using a pseudo-time variable $t(\epsilon) = \epsilon_{\text{init}} - \epsilon$ the latter term in Equation (A.56) becomes

$$-S\partial_\epsilon\psi = -S\frac{\partial t}{\partial\epsilon}\partial_t\psi = S\partial_t\psi. \tag{A.57}$$

For the P_N-model the source term in StaRMAP is set to zero $Q = 0$. Boundary conditions for the P_N model are given in Equation (A.30) and comply with the definition of StaRMAP's generic moment Equation (A.33). Hence, $M^{(n)}_{e,o} = L^{(n)}_o A^{(n)}_{e,o}$.

These transformations allow us to define the partial differential equation which is solved to compute the moments of the electron fluence.

$$\begin{aligned} &-S\partial_\epsilon\psi + A^{(x)}\partial_x\psi + A^{(y)}\partial_y\psi + A^{(z)}\partial_z\psi + (-\partial_\epsilon S + C)\psi = 0 \quad \forall\epsilon \in [\epsilon_{\text{cut}}, \epsilon_{\text{init}}], x \in \mathfrak{S} \\ &\quad \psi(\epsilon = \epsilon_{\text{init}}, x) = 0 \quad \forall x \in \mathfrak{S} \\ &\quad \psi_{o,n}(\epsilon, x) = L^{(n)}_o A^{(n)}_{o,e}\psi_{e,n}(\epsilon, x) + g_{o,n}(\epsilon, x) \quad \forall\epsilon \in [\epsilon_{\text{cut}}, \epsilon_{\text{init}}], x \in \partial\mathfrak{S}. \end{aligned} \tag{A.58}$$

Additionally, we define an operator which computes one step of the P_N-equation using the discretization introduced in Section A.2.1

$$\psi_{i+1} = \psi(\epsilon_{n_\epsilon - i}) = P_N^{i\to i+1}(\psi_i, \rho). \tag{A.59}$$

The P_N-equation is solved "backwards" in energy, hence the definition of the $(i+1)$th solution $\psi_{i+1} = \psi_{\epsilon_{n_\epsilon}-i}$, where n_ϵ is the number of energy steps, $\epsilon_1 = \epsilon_{\text{cut}}$ the smallest (cutoff) energy and $\epsilon_{n_\epsilon} = \epsilon_{\text{init}}$ the highest (initial) energy. The operator $P_N^{i\to i+1}$ approximates the material properties S and C at the intermediate energy $\frac{\epsilon_i+\epsilon_{i+1}}{2}$ using the mass concentrations ρ, computes boundary values as described in Equation (A.53) and performs an update of the moments ψ using Strang Splitting as defined in Equation (A.50).

A.2.3 Numerical Integration: Ionization Distribution

For the integration (Equation (A.10)) in energy we apply the trapezoidal rule

$$\int_{\epsilon_{\text{cut}}}^{\epsilon_{\text{init}}} \sigma_{(Z,j)}(\epsilon)\psi_0^0(\epsilon,x)\,\mathrm{d}\epsilon \approx \sum_{i=1}^{n_\epsilon-1} \Delta\epsilon_i \frac{\sigma_{(Z,j)}(\epsilon_{i+1})\psi_0^0(\epsilon_{i+1},x) + \sigma_{(Z,j)}(\epsilon_i)\psi_0^0(\epsilon_i,x)}{2}, \tag{A.60}$$

where $\Delta\epsilon_i = \epsilon_{i+1}-\epsilon_i$. For an implementation, which can be performed analogously to the time stepping of the P_N-equation using StaRMAP without storing the solution at the previous time step, we can rewrite the sum to

$$\frac{\Delta\epsilon_1\sigma_{(Z,j)}(\epsilon_1)\psi_0^0(\epsilon_1,x)}{2} + \sum_{i=2}^{n_\epsilon-1}\left(\frac{\Delta\epsilon_{i-1}+\Delta\epsilon_i}{2}\sigma_{(Z,j)}(\epsilon_i)\psi(\epsilon_i,x)\right) + \frac{\Delta\epsilon_{n_\epsilon}\sigma_{(Z,j)}(\epsilon_{n_\epsilon})\psi_0^0(\epsilon_{n_\epsilon},x)}{2}. \tag{A.61}$$

A.2.4 Numerical Integration: Mass Attenuation

The x-ray generation distribution $\mathcal{X}_{(Z,j)}$ is like the solution ψ_0^0 discretized on the grid G_{eee}. For each spatial coordinate $x_{i,j,k} \in G_{eee}$, we approximate the attenuation factor using the trapezoidal rule for the line integral in Equation (A.7)

$$\exp\left(-\int_{d(x)} \mu_{(Z,j)}(\bar{z})\,\mathrm{d}\bar{z}\right) \approx \exp\left(-\sum_{\bar{z}_m\in\bar{d}(x)} \mathbb{t}_m\mu_{(Z,j)}(\bar{z}_m)\Delta\bar{z}\right), \tag{A.62}$$

where $\bar{d}(x) = \{\bar{z}_m | m = 1,\dots,n_{\mathcal{A}}\}$ is a discretization of the path $d(x)$ into $n_{\mathcal{A}}$ points with distance $\Delta\bar{z}$ and $\mathbb{t}_i = 1 - \frac{1}{2}(\delta_{i,1}+\delta_{1,n_{\mathcal{A}}})$ the factor for the trapezoidal rule. Then the formula for the approximation of the detector intensity is

$$\mathcal{I}_{(Z,j)} = \mathcal{A}_{(Z,j)} \approx \sum_{x\in G_{eee}} \exp\left(-\sum_{\bar{z}_m\in\bar{d}(x)} \mathbb{t}_m\mu_{(Z,j)}(\bar{z}_m)\Delta\bar{z}\right)\mathcal{X}_{(Z,j)}(x)\mathrm{vol}(\Delta x), \tag{A.63}$$

where $\mathrm{vol}(\Delta x)$ is the spatial integration volume.

A.3 Parametrization of the Material

As discussed in Section 2.2 the particular choice of a model of mass concentrations ρ is problem dependent and should be conducted with great care and a particular material in mind.

In this section we provide models for mass concentrations which are utilized to conduct the reconstruction experiments in Section B.3. Thereby we distinguish between mass concentration models (Section A.3.1) and parametrizations (Section A.3.3). The former is the relation of mass concentrations with other physical quantities and the latter is the parametrization of a function in 3D. Combining a mass concentration model with a parametrization yields the relation

$$\rho(x; p) : \mathbb{R}^3 \times \mathbb{R}^{n_p} \to \mathbb{R}^{n_e}, \tag{A.64}$$

a parametrization of the function $\rho(x)$ which returns mass concentrations of n_e elements given a position x inside the material and n_p parameters, which describe the material.

A.3.1 Relations of Mass Concentrations

Recall the definition of mass concentrations from Section A.1.1. We drop the dependency $\cdot(x)$ because all following quantities refer to the point x, respectively the volume dx.

A.3.1.1 Relation To Mass Fractions

The mass fraction $\omega_i = \frac{M_i}{M_{\text{tot}}}$ of an element i in a compound is defined as the ration between partial mass M_i and total mass M_{tot}. We assume, that the compound has been formed in a volume-preserving manner (the volume of the compound is the sum of volumes of pure constituents). The assumption gives $V = \sum_{i=1}^{n_e} \frac{M_i}{\rho_i^{\text{pure}}}$, where ρ_i^{pure} is the density of pure constituents, and we find

$$\rho_i = \frac{M_i}{V} = \frac{M_i}{\sum_{i=1}^{n_e} V^{\text{pure}}} = \frac{M_i}{M_{\text{tot}} \sum_{j=1}^{n_e} \frac{\omega_i}{\rho_j^{\text{pure}}}} = \omega_i \left(\sum_{j=1}^{n_e} \frac{\omega_j}{\rho_j^{\text{pure}}} \right)^{-1}. \tag{A.65}$$

A relation between mass concentrations ρ_i and mass fractions ω_i.

A.3.1.2 Relation To Volume Fractions

The volume fraction $\varphi_i = \frac{V_i}{V}$ of element i in a compound is defined by the ration between partial volume V_i and total volume V_{tot}. If the density of element i in the compound is the same as if the constituent i would be pure $\frac{M_i}{V_i} = \rho_i^{\text{pure}}$, we can derive the following relation of volume fractions φ_i and mass concentrations ρ_i. Inserting the assumption into the definition of mass concentrations Equation (A.2) yields

$$\rho_i = \frac{M_i}{V} = \frac{V_i \rho_i^{\text{pure}}}{V} = \varphi_i \rho_i^{\text{pure}}. \tag{A.66}$$

A.3.1.3 Relation To a Linear Density Model

We assume that the density of the compound can be modeled by linear combination of densities of pure constituents $\rho_{\text{tot}} = \sum_{i=1}^{n_e} \gamma_i \rho_i^{\text{pure}}$, and that the mass concentrations are the

γ_ith fraction of the total density. Combined this gives a quadratic relation between γ_i and ρ_i

$$\rho_i = \gamma_i \rho_{\text{tot}} = \gamma_i \sum_{j=1}^{n_e} \gamma_j \rho_j^{\text{pure}}. \tag{A.67}$$

A.3.1.4 Additional Constraints of Mass Concentration Models

By their definition in Equation (A.2), the mass concentrations are constrained by

$$\rho_{\text{tot}} = \sum_{i=1}^{n_e} \rho_i. \tag{A.68}$$

However, the total density ρ_{tot} might not be known or might vary inside the material, thus the constraint cannot be used in practice. Assuming a model for the mass concentrations, the constraint is inherited to mass fractions and volume fractions, which can be constrained by $\sum_{i=1}^{n_e} \{\omega, \varphi, \gamma\}_i = 1$. Hence, the additional assumptions allow us to reduce the parameter space by enforcing

$$\begin{gathered} 0 \leq \{\omega, \varphi, \gamma\}_i \qquad \sum_{i=1}^{n_e-1} \{\omega, \varphi, \gamma\}_i \leq 1 \\ \{\omega, \varphi, \gamma\}_{n_e} = 1 - \sum_{i=1}^{n_e-1} \{\omega, \varphi, \gamma\}_i. \end{gathered} \tag{A.69}$$

The parameter of the last element $\{\omega, \varphi, \gamma\}_{n_e}$ can always be computed from the previous ones.

A.3.2 Comparison of Modeled Mass Concentrations

In Figure A.4 the mass concentration models are compared for a binary material consisting of lead ($\rho_{Pb}^{\text{pure}} = 11.34\,\text{g}\,\text{cm}^{-3}$) and silicon ($\rho_{Si}^{\text{pure}} = 2.329\,\text{g}\,\text{cm}^{-3}$). For each of the three models, we vary the parameter for lead $\{\omega, \varphi, \gamma\}_{Pb}$ while the parameter for silicon is given by $\{\omega, \varphi, \gamma\}_{Si} = 1 - \{\omega, \varphi, \gamma\}_{Pb}$. We observe equivalence in the prediction of the total density using the volume fraction and the linear density model, however, the mass concentrations differ in all models. A general statement about quality for a particular model lies beyond our knowledge and is probably material dependent.

From the mathematical perspective the choice of a proper material model does have an impact on the invertibility (see Section B.3.1)

A.3.3 Parametrizations

In this section we propose some parametrizations for the discretization of a multivariate 3D function. Although we visualize the concepts only in 2D, the extension to 3D or the reduction to 1D is straightforward. We denote the parametrized function with $P(x; p) : \mathbb{R}^3 \times \mathbb{R}^{n_p} \to \mathbb{R}^{n_o}$ and its parameters with $p \in \mathbb{R}^{n_p}$. We denote the number of output values of the parametrization P with n_o. For the sake of comparability, we precisely formulate the number of required parameters n_p for each parametrization.

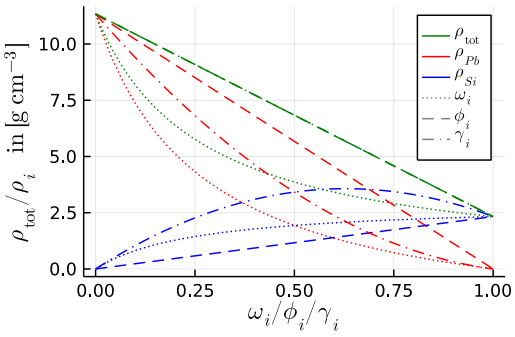


Figure A.4: *Mass concentrations ρ_i and total density ρ_{tot} of a compound consisting of lead Pb and silicon Si, derived from mass fraction ω_i, volume fractions φ_i and γ_i.*

A.3.3.1 Piecewise-Constant Parametrization

Assume that the reconstruction domain is split into several cuboids(3D)/ rectangles(2D) (see Figure A.5), with aligned interfaces in all dimensions. We specify a rectangular box by

$$\mathcal{B}_{i,j,k} = \{(x,y,z) \in \mathbb{R}^3 | x_i < x \leq x_{i+1}, y_j < y \leq y_{j+1}, z_k < z \leq z_{k+1}\}, \tag{A.70}$$

where the interfaces are given by $\{x_i\}_{i=1,\dots,n_x}$, $\{y_j\}_{j=1,\dots,n_y}$ and $\{z_k\}_{k=1,\dots,n_z}$ with n_x, n_y and n_z the number of interfaces in each dimension. Based on the subsets, we define the 3D piecewise-constant parametrization by

$$P(x;p) = \begin{cases} p_{i,j,k} & x \in \mathcal{B}_{i,j,k} \\ p_{\text{out}} & \text{else} \end{cases}. \tag{A.71}$$

In each of the boxes $\mathcal{B}_{i,j,k}$ the parametrization P is constant with a value defined by the parameters $p_{i,j,k}$. If x happens to lie outside all the boxes, we assign p_{out}. The number of parameters is given by

$$n_p = ((n_x - 1)(n_y - 1)(n_z - 1) + 1)n_o. \tag{A.72}$$

Because p_{out} counts as an additional "box", we add 1. Note that the positions of the interfaces x_i, y_j and z_k are fixed and not part of the parameters.

A.3.3.2 Linear Parametrization

The linear parametrization is also based in the splitting of the reconstruction domain into several boxes (see Figure A.5). But instead of specifying the value of P inside each box, we specify the value of P on the vertices of the boxes. We define the linear interpolation by

$$P(x,p) = \begin{cases} \mathcal{I}(x; p_{i,j,k}, p_{i+1,j,j}, p_{i,j+1,k}, \dots, p_{i+1,j+1,k+1}) & x \in \mathcal{B}_{i,j,k} \\ \mathcal{I}(x; p_{\text{out}}) & else \end{cases} \tag{A.73}$$

where $\mathcal{I}$ denotes 3D linear interpolation of the values on the vertices.

We visualize the linear interpolation in 2D in Figure A.6 and derive the interpolation formula. Given a point (x,y) which is inside the Rectangle $\{(x,y)|x_- \leq x < x_+, y_- \leq y < y_+\}$, we derive interpolation weights η_x and η_y

$$\eta_x = \frac{dx}{Dx} = \frac{x - x_-}{x_+ - x_-} \quad \eta_y = \frac{dy}{Dy} = \frac{y - y_-}{y_+ - y_-}. \tag{A.74}$$

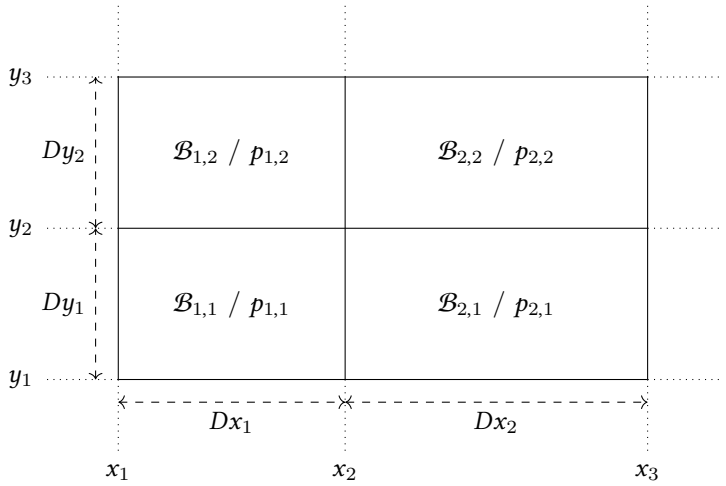


Figure A.5: *2D sketch of the piecewise-constant parametrization. Interfaces of the boxes $\mathcal{B}_{i,j}$ are defined by x_i and y_j. The boxes are not required to be equally sized and their side lengths are defined by Dx_i and Dy_i. Inside each box $\mathcal{B}_{i,j}$ the function value is constant $P(x \in \mathcal{B}_{i,j}) = p_{i,j}$.*

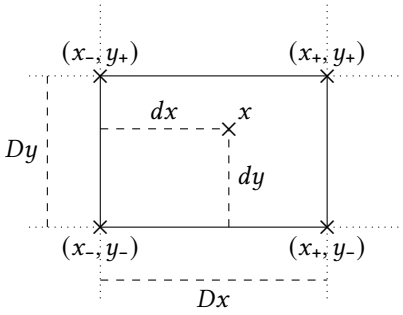


Figure A.6: *2D sketch of linear parametrization inside a box with vertices (x_-, y_-), (x_+, y_-), (x_+, y_+) and (x_-, y_+). Distances of x from the interfaces are denoted with dx resp. dy and the side lengths of the box with Dx and Dy.*

Based on the parameters defined on the vertices p_{--}, p_{+-}, p_{-+} and p_{++} the interpolation is given by

$$\begin{aligned}\mathcal{I}(x; p_{--}, p_{+-}, p_{-+}, p_{++}) =& (1-\eta_x)(1-\eta_y)p_{--} \\ &+ \eta_x(1-\eta_y)p_{+-} \\ &+ (1-\eta_x)\eta_y p_{-+} \\ &+ \eta_x\eta_y p_{++}.\end{aligned} \tag{A.75}$$

A linear interpolation using n_x, n_y and n_z interfaces has

$$n_p = (n_x n_y n_z + 1)n_o \tag{A.76}$$

parameters. We add 1 if we additionally specify a value for points outside all interfaces.

A.3.3.3 Neural Network Parametrization

The previous parametrizations are characterized by a predefined geometry, which limits their approximation capabilities when trying to represent a general function. The only possibility to increase their flexibility is by increasing the number of boxes in each dimension, which simultaneously increases the number of parameters (in 3D: cubed).

Another possible function parametrization is based on neural networks [Bishop, 2006; Goodfellow et al., 2016]. The parameters in a neural network simultaneously parametrize the geometry of the function and the function value. Neural networks are universal function approximators [Leshno et al., 1993; Iserles, 1999], hence if the number of parameters is large enough, they are able to precisely approximate any continuous function. The same statement holds for the piecewise-constant parametrization as well as for the linear parametrization, however, neural networks are applied in a wide variety of applications and have proven to be suitable for diverse tasks.

The basic structure of feed-forward neural networks consists of multiple layers k, with a defined number of inputs $x^k \in \mathbb{R}^{n^k_{\text{in}}}$ and number of outputs $z^k \in \mathbb{R}^{n^k_{\text{out}}}$ for each layer. A dense layer is defined by the mapping

$$z^k = \sigma^k(W^k x^k + b^k), \tag{A.77}$$

where σ^k is an activation function, $W^k \in \mathbb{R}^{n^k_{\text{out}} \times n^k_{\text{in}}}$ and $b_k \in \mathbb{R}^{n^k_{\text{out}}}$ are the weights and bias of layer k. The layers are connected by $x_{k+1} = z_k$; the output of layer k is the input of layer $k+1$. This defines the condition, that the number of outputs of layer k and the number of inputs of layer $k+1$ has to coincide, $n^k_{\text{out}} = n^{k+1}_{\text{in}}$. Common choices for the activation function are $\sigma^k \in \{\tanh, \text{ReLU}, \text{sigmoid}, \text{id}, \ldots\}$.

For our use case of neural networks, two further assumptions on the dimensions of the layers are necessary. The number of inputs is given by the number of spatial dimensions $n^1_{\text{in}} = 3$ and the number of outputs is specified by the number of considered constituents of the material.

Per layer the neural network has $n^k_p = n^k_{\text{in}} n^k_{\text{out}} + n^k_{\text{out}}$ parameters, therefore the number of parameters is

$$n_p = \sum_k n^k_{\text{in}} n^k_{\text{out}} + n^k_{\text{out}}. \tag{A.78}$$

We describe two additional layers, which we used during our analysis.

Softmax layer The softmax layer is often used to interpret its outputs as probabilities, because it assures that $\sum_j z_j = 1$. The layer does not introduce any new parameters and is given by

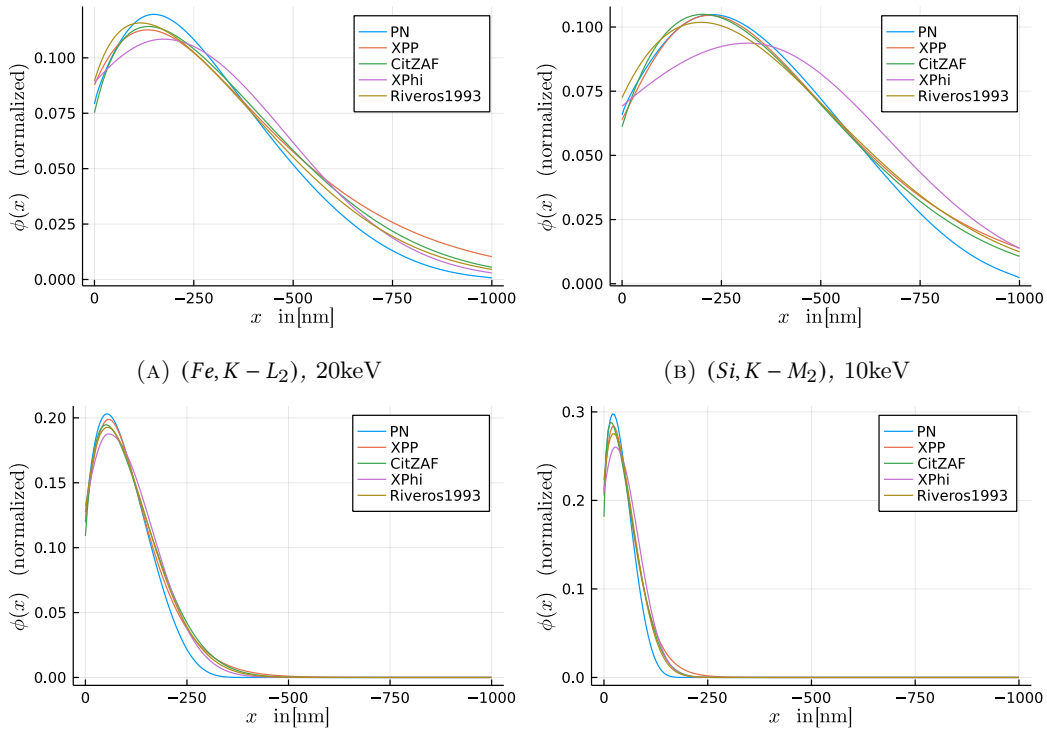
$$z_i = \frac{\exp(x_i)}{\sum_j \exp(x_j)}. \tag{A.79}$$

In the context of a material parametrization, we can use a softmax layer to automatically guarantee compliance to the constraints derived in Section A.3.1.4.

Normalization Layer Algorithms for neural networks are usually designed to operate with normalized inputs and outputs. If we analyze function variations on the nm scale in the inputs a proper scaling is necessary. We transformed the spatial input variables to have

zero-mean and a variance of one. This can be implemented by a linear transformation (a dense layer with id activation function, but with fixed parameters) which transforms $\mathfrak{S}$ to $(-1, 1)^3$.

For the numerical results of the forward model in Section A.4 and the reconstruction experiments in Section B.3 the parametrizations are reduced to 2D or 1D equivalents. In Section C.2 we describe an extension of the neural network parametrization which specifically models an elliptical material structure.



(A) $(Fe, K - L_2)$, 20keV

(B) $(Si, K - M_2)$, 10keV

(C) $(Cu, L_1 - M_2)$, 10keV

(D) $(Ga, L_1 - M_2)$, 5keV

Figure A.7: *Comparison of $\phi(\rho z)$ curves computed using the presented method (blue, PN) to classical models (colored, XPP, CitZAF, XPhi and Riveros). The ionization distribution $\phi_{(Z,j)}$ in homogeneous materials is compared for multiple x-rays using different beam energies ($(Fe, K - L_2)$ $\mu_\epsilon = 20$keV; $(Si, K - M_2)$ $\mu_\epsilon = 10$keV; $(Cu, L_1 - M_2)$ $\mu_\epsilon = 10$keV; and $(Ga, L_1 - M_2)$ $\mu_\epsilon = 5$keV). The underlying electron fluence is approximated using the P_{21} model.*

A.4 Numerical Experiments - Forward Model

A.4.1 Comparison: Ionization Distribution with Classical Phi-Rho-Z Models as Reference

We compare the ionization distribution $\phi_{(Z,j)}$ for homogeneous materials computed using the presented method to classical analytical $\phi(\rho z)$ models as reference. We compare against the XPP [Pouchou and Pichoir, 1991], CitZAF [Heinrich and Newbury, 1991], XPhi [Llovet and Merlet, 2010] and Riveros [Riveros and Castellano, 1993] models, which are implemented in the library `NeXLCoreMatrixCorrection.jl` [Ritchie, 2021b]. Classical $\phi(\rho z)$ models describe the ionization distribution ϕ over the mass depth ρz, where ρ is the density of the material and z is the spatial coordinate for depth. The other spatial dimensions x and y are neglected. Hence, it is sufficient to employ a reduced form of the developed 3D P_N model where only the depth is considered. The spatial domain $\mathfrak{S} = (-1000\text{nm}, 0\text{nm})$ is discretized on 200 grid nodes in depth (x-direction in our model). We compare the ionization distribution $\phi_{(Z,j)}$ for multiple beam energies μ_ϵ with standard deviation $\sigma_\epsilon = 0.1$keV. The spatial

and directional distribution of the beam does not influence the reduced 1D P_N model. We choose $\epsilon_{\text{init}} = \mu_\epsilon + 1\text{keV}$ and $\epsilon_{\text{cut}} = \min_{(Z,j)} \epsilon^{\text{edge}}_{(Z,j)} - 0.1\text{keV}$, such that the beam is sufficiently captured and the minimal edge energy of all x-rays is included. The edge energy ϵ^{edge} is the minimal energy where electrons are able to ionize Z atoms. We define materials which are homogeneous in depth and only consist of a single element Z. Hence, the mass concentration in the whole material is constant $\rho_Z(x) = \rho_Z^{\text{pure}}$. In Figure A.7 we present comparisons of the $\phi_{(Z,j)}(x)$-curves for multiple materials and x-rays. Although our model tends to have a slightly smaller ionization depth for all illustrated curves, the agreement is satisfactory.

A.4.2 Showcase: 1D Electron Fluence and Ionization Distribution for Layered Coatings

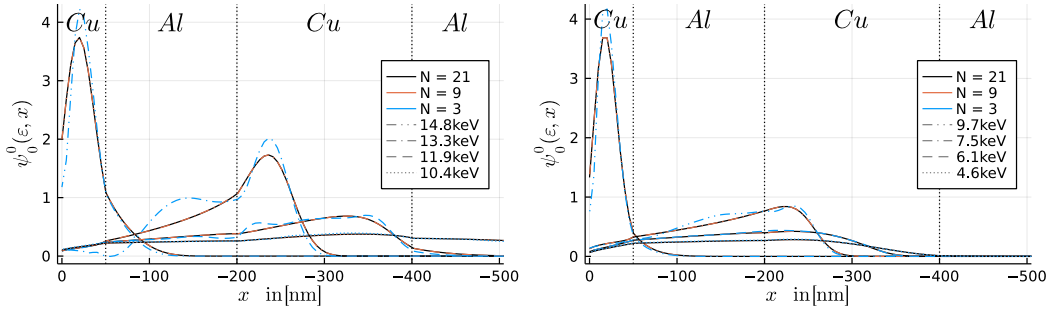
We showcase the capability of the presented model to compute the electron fluence $\psi_0^0(\epsilon, x)$ and the ionization distributions $\phi_{(Z,j)}(x)$ for a more complex material. In depth the material consists of multiple layers of homogeneous copper *Cu* and Aluminum *Al* that are of variable thickness. Beginning from the surface: a 50nm *Cu*-layer, a 150nm *Al*-layer, a 200nm *Cu*-layer, a 300nm *Al*-layer succeeded by *Cu*.

We investigate two different electron beam energies, 15keV and 10keV, both with a variance of $\sigma_\epsilon = 0.2\text{keV}$. Again, the spatial and directional distribution of the beam is not required, because we only consider the 1D P_N model. The computational domain for both experiments is $\mathfrak{S} = (-1000\text{nm}, 0\text{nm})$ and the considered energies are $[\epsilon_{\text{cut}}, \epsilon_{\text{init}}] = [1\text{keV}, 15.5\text{keV}]$.

For comparison of different approximation qualities using a different number of moments P_N, we approximate the moments of the electron fluence using P_3, P_9 and P_{21}. In Figure A.8 the zeroth moment of the electron fluence $\psi_0^0(\epsilon, x)$ is plotted over the depth x for the 15keV beam (Figure A.8a) and the 10keV beam (Figure A.8a). Blue (P_3), orange (P_9) and black (P_{21}) lines differentiate between the moment order of the underlying expansion. Additionally, the material interfaces are indicated (by gray vertical lines) and labeled. The high order approximations P_9 and P_{21} coincide, but differences in the electron fluence for P_3 are clearly visible. Especially in the vicinity of interfaces, where the low order moment approximation (blue P_3) shows oscillating behavior.

The differences for the beam energies are obvious. Because of the smaller energy, the zeroth moment of the electron fluence ψ_0^0 for the 10keV beam decays closer to the surface, while ψ_0^0 for the 15keV beam reaches deeper layers of the material.

In Figure A.9 we present normalized ionization distribution curves which are computed by integration of the presented electron fluence in energy (c.f. Equation (A.10)). Note that the ionization distribution ϕ is the probability of ionization if a respective atom would be present at the given location, hence e.g. the ionization probability $\phi_{(Z,j)}$ for the x-ray characteristic to an element Z can be non-zero in a layer even if Z is not present in that layer. We collect curves for the $(Al, K - L_2)$ and the $(Cu, K - L_2)$ x-rays for both beam energies. Ionization curves drawn in dashed lines base on the P_3 electron fluence approximation while solid lines base on the more precise P_{21} approximation. The P_9 ionization curve coincides with the P_{21} curve and is visualized by black dots. The interfaces as well as the beam energies (indicated by vertical dotted lines) show an undeniable effect on the ionization distribution. While for the high energy beam, *Al* atoms would be ionized in the first five layers, the low



(A) ψ_0^0 *for an electron beam of* 15keV. (B) ψ_0^0 *for an electron beam of* 10keV.

Figure A.8: *The zeroth moment of the electron fluence* $\psi_0^0(\epsilon, x)$ *in a material, which consists of multiple* Cu *and* Al *layers in depth. The electron fluence is computed using three spherical harmonic expansion orders,* P_3*,* P_9 *and* P_{21}*, and is plotted for all at multiple energies* ϵ*. The simulation is conducted for two different beam energies. The curves of* P_9 *and* P_{21} *coincide, hence the orange (*P_9*) curve overlaps the black (*P_{21}*) curves. Note the varying energy values* ϵ *for each plot.*

energy beam only ionizes the first three layers. Ionization of Cu atoms using the low energy beam is almost impossible, because the beam energy μ_ϵ is only marginally higher than the edge energy of Cu.

Note that in comparison to the classical $\phi(\rho z)$ curves for materials with thin layers, our ionization distribution for layered materials does not base on the sophisticated concatenation of $\phi(\rho z)$ curves of homogeneous materials [Pouchou and Pichoir, 1991]. Our ionization distribution solely depends on the underlying approximation of the electron fluence ψ_0^0.

A.4.3 Showcase: 2D Electron Fluence and Ionization Distribution for a Copper-Silicon Material

We showcase the 2D computation of the electron fluence and the ionization distribution for a material consisting of copper Cu and silicon Si. In Figure A.10 the structure of the material is visualized. The computational domain $\mathfrak{S} = (-500\text{nm}, 0\text{nm}) \times (-300\text{nm}, 300\text{nm})$ is split at $y = 0\text{nm}$ into a left and a right section. In depth (x) each section consists of a surface layer of 100nm and a substrate. The homogeneous Cu and Si structure in each section of the material is reversed.

The beam hits the material at $\mu_x = (0\text{nm}, 0\text{nm})$ (with standard deviation $\sigma_x = 30\text{nm}$) into negative x direction (with concentration coefficient $\kappa = 10$) and has an energy of $\mu_\epsilon = 15\text{keV}$ (with standard deviation $\sigma_\epsilon = 0.2\text{keV}$). The 2D computational domain $\mathfrak{S}$ is discretized by 150×150 spatial coordinates and for the energy interval we choose $[15.5\text{keV}, 1.0\text{keV}]$.

In Figure A.11 we compare the zeroth moment of the electron fluence ψ_0^0 for multiple electron energies and different P_N expansion orders (P_3, P_9 and P_{21}). For a low order approximation (P_3) artifacts of the method are clearly visible. Especially at material interfaces, the method tends to produce oscillations and even computes negative values for the electron fluence. In the presented plots, the color value is computed by clipping negative values to zero. Increasing the order of the moment approximation, decreases the pronouncedness of the artifacts.

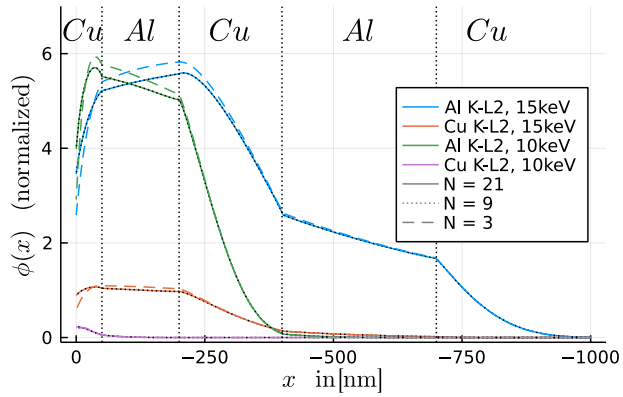


Figure A.9: *The ionization distribution ϕ for $(Al, K-L_2)$ and $(Cu, K-L_2)$ in a 1D material which is layered in depth. ϕ is based on the electron fluence computed from two beam energies* 15keV *and* 10keV *as well as three moment approximation orders P_3, P_9 and P_{21}. The curves of P_9 and P_{21} coincide, hence the P_9 curves are visualized by black dots.*

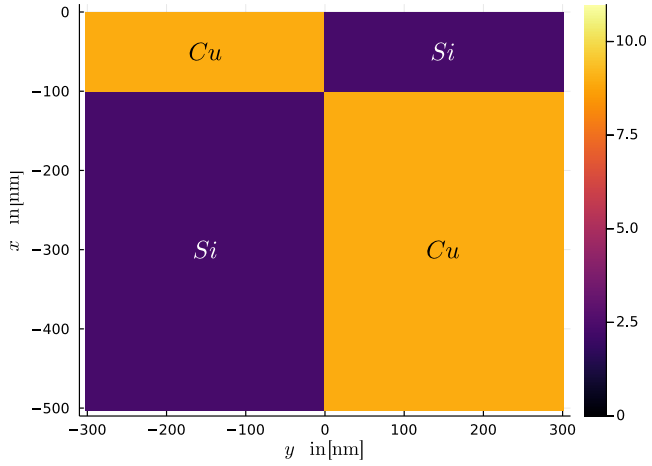
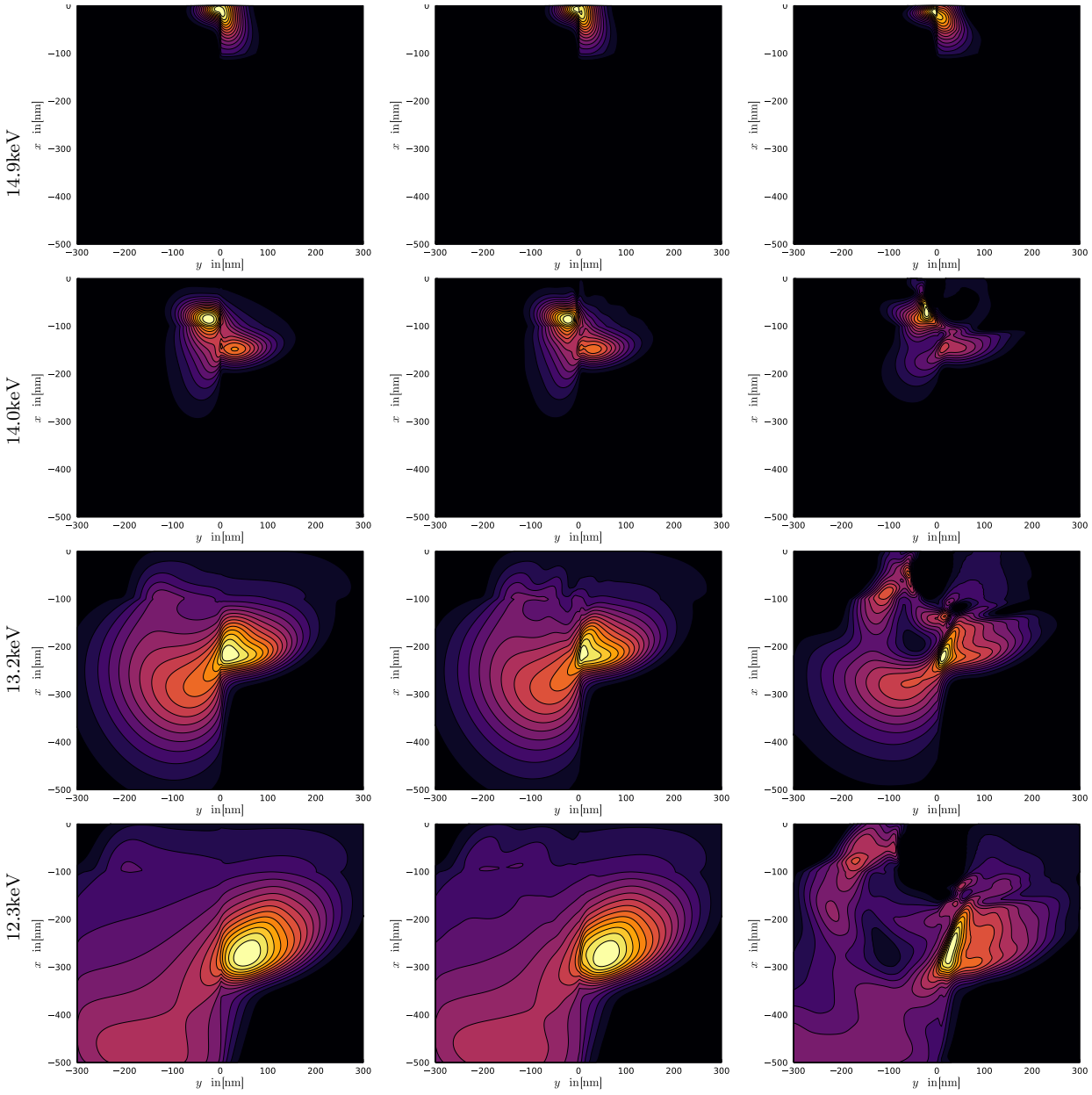
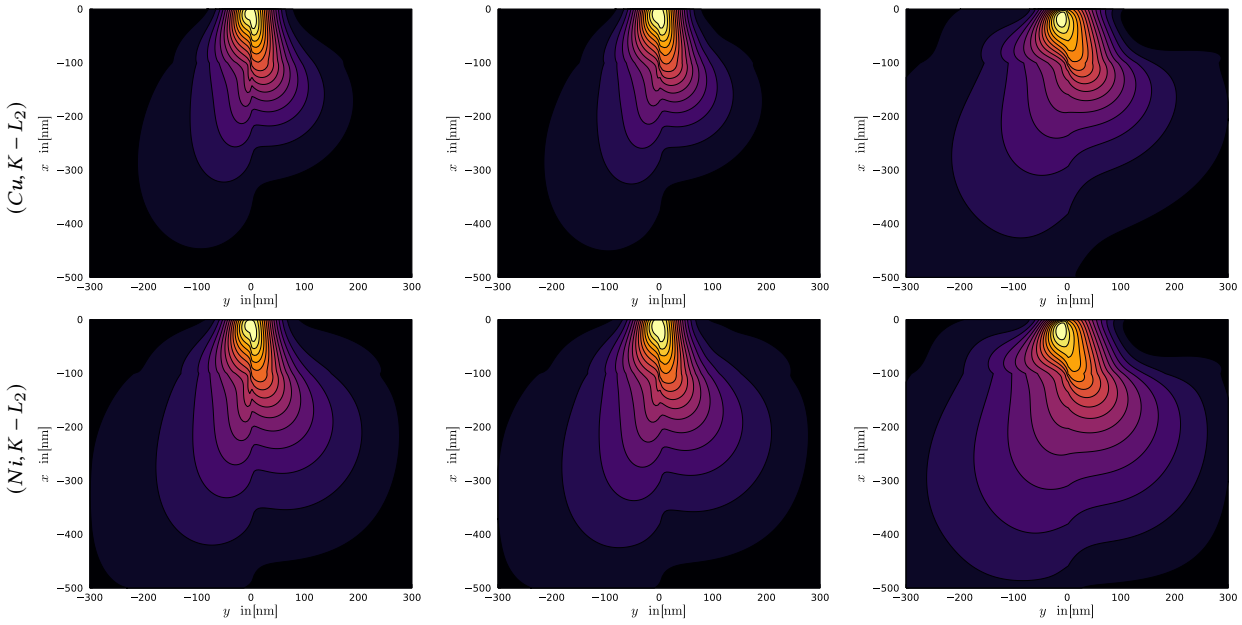


Figure A.10: *The material structure used to showcase the 2D electron fluence and ionization distribution. The material is layered in depth* x *and width* y *with an alternating pattern of* Cu *and* Si*. The color values represent the density $\rho(x)$ of the material in* $[\mathrm{g\,cm^{-3}}]$*. Note that* Si *is less dense than* Cu*, hence the interfaces influence the electron transport significantly.*



(A) P_{21} (B) P_9 (C) P_3

Figure A.11: *The zeroth moment of the 2D electron fluence* ψ_0^0 *computed using a different number of moments* $P_N \in \{P_{21}, P_9, P_3\}$. *The domain is discretized in* 150×150 *spatial coordinates. The beam is centered at* $y = 0\text{nm}$, $\sigma_x = 30\text{nm}$ *with energy* 15keV, $\sigma_\epsilon = 0.2\text{keV}$ *in negative* x*-direction* $(-1, 0)$, $\kappa = 10$.



(A) P_{21} (B) P_9 (C) P_3

Figure A.12: *The ionization distribution ϕ for $(Cu, K-L_2)$ and $(Si, K-L_2)$. The underlying electron fluence is computed using multiple P_N models (c.f. Figure A.11). Note that the ionization distribution is the probability of ionization assuming that the respective atoms are present, hence e.g. $\phi_{(Cu,K-L_2)}$ is nonzero even if $\rho_{Cu}(x) = 0$.*

In Figure A.12 we illustrate the ionization distribution fields $\phi(x)$ for the x-rays $(Cu, K-L_2)$ and $(Si, K-L_2)$, based on the approximations of the electron fluence using the P_N-model. Due to the integration in Equation (A.10), the artifacts of the low order approximation P_3 are smoothed and no longer visible. Still, the ionization distribution based on the P_3 approximation differs from the higher order approximations P_9 and P_{21}.

A.4.4 Showcase: 3D Electron Fluence

We showcase the 3D computation of the electron fluence for a material consisting of copper Cu and nickel Ni. The structure of the material is similar to the previous showcase (100nm surface layer), but with 2 additional sections in z-direction. The computational domain is $\mathfrak{S} = (-300\text{nm}, 0) \times (-200\text{nm}, 200\text{nm}) \times (-200\text{nm}, 200\text{nm})$, thus smaller than in the previous section, because copper and nickel have a higher density than silicon, hence a smaller interaction volume. The beam setup is: $\mu_x = (0\text{nm}, 0\text{nm})$, $\sigma_x = 10\text{nm}$, $\mu_\epsilon = 12\text{keV}$, $\sigma_\epsilon = 0.3\text{keV}$, $\mu_\Omega = [-1, 0, 0]$, $\kappa = 10$. We discretize the spatial domain in $50 \times 50 \times 50$ spatial coordinates and choose $[12.5\text{keV}, 8.3\text{keV}]$ as the energy interval.

In Figure A.13 the zeroth moment of the electron fluence ψ_0^0 is visualized for multiple electron energies $\{12, 11, 10, 9\}\text{keV}$. Note that the densities of copper and nickel are very similar, hence changes in stopping power and transport coefficient at the material interfaces are marginal and no difference in the electron fluence is visible.

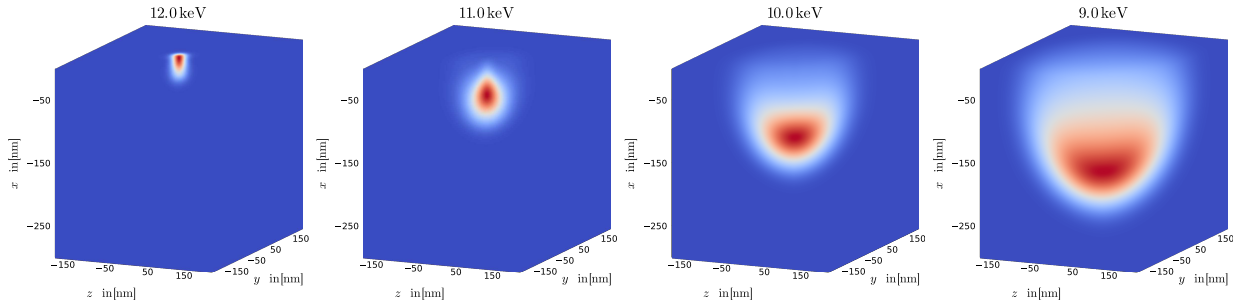


Figure A.13: *The zeroth moment of the 3D electron fluence* ψ_0^0 *computed using* P_9 *in a material consisting of copper* Cu *and nickel* Ni*. The domain is discretized in* $50 \times 50 \times 50$ *spatial coordinates. The beam is centered at* $(z, y) = (0\text{nm}, 0\text{nm})$*,* $\sigma_x = 10\text{nm}$ *with energy* $\epsilon = 12\text{keV}$*,* $\sigma_\epsilon = 0.3\text{keV}$ *in negative* x*-direction* $[-1, 0, 0]$*,* $\kappa = 50$*.*

Chapter B

The Inverse Problem

In Chapter A we developed and discussed a model that is able to predict k-ratios for complex material samples. At all times we assumed the material to be given. However, the actual challenge in EPMA is the unveiling of the material from given k-ratio measurements. The remainder of this thesis (Chapter B) deals with the inverse problem, the reconstruction of the material; specifically with the generic computation of gradients through the forward model presented in Chapter A. The generic computation of gradients means:

- we allow arbitrary (differentiable) computations on top of the modeled k-ratios; we remain generic in the objective function;
- we allow arbitrary (differentiable) computations below mass concentrations; we remain generic in the parametrization of the material.

Chapter B is divided in Sections B.1 to B.3. In Section B.1 we build on the ideas from Section 2 and further motivate the aim of remaining general. Section B.2 then details on the applied methods and concepts to achieve this generality. We conclude Chapter B with a validation of the implemented method and reconstruction experiments.

B.1 Reconstruction as an Optimization Problem

The formulation of the reconstruction as an optimization problem mainly consists of the following steps: The definition of the parameters p which are searched for, the definition of the forward model $k(p)$ and the definition of the objective

$$\mathrm{disc}(k(p), k^{\mathrm{exp}}) \tag{B.1}$$

as some measure of discrepancy between modeled $k(p)$ and measured k^{exp} k-ratios. The reconstruction result is then the set of parameters p^* for which the modeled k-ratios have the smallest discrepancy from the measured k-ratios.

$$p^* = \arg\min_p \mathrm{disc}(k(p), k^{\mathrm{exp}}) \tag{B.2}$$

Usually the discrepancy $\mathrm{disc}(\cdot,\cdot)$ denotes a norm, therefore $\mathrm{disc}(\cdot,\cdot) \geq 0$. However, we cannot expect to find p^* such that $\mathrm{disc}(k(p^*), k^{\mathrm{exp}}) = 0$ in a real reconstruction problem, because of imperfect modeling and the presence of noise.

Classical reconstruction approaches, e.g. matrix correction methods for homogeneous materials or layered specimen based on $\phi(\rho z)$ curves, define homogeneous mass concentrations or the mass thickness of layers as the parameters p. The k-ratio model either bases on multiplicative correction factors or the integration of $\phi(\rho z)$ curves. The discrepancy $\mathrm{disc}(\cdot,\cdot)$ is defined as the squared error of modeled and measured k-ratios

$$\mathrm{disc}(k(p), k^{\mathrm{exp}}) = \sum_{(Z,j)} (k_{(Z,j)}(p) - k^{\mathrm{exp}}_{(Z,j)})^2. \tag{B.3}$$

The use of the squared error as the objective function is not limited to matrix correction methods and can be used analogously with our model.

Defining the squared error as the objective function is very common, but may feel arbitrary. Inverse problem theory justifies the choice of the squared error function by deriving it from statistical assumptions [Tarantola, 2005]. The assumptions simultaneously allow a statistical interpretation of the reconstruction result p^*.

In Section 2.1, we mentioned the approximation of maximum likelihood (ML) and maximum posterior estimates (MAP). The maximum likelihood estimate p^* is

$$p^* = \arg\max_p \pi(k^{\mathrm{exp}}|p), \tag{B.4}$$

the maximum of the probability density function of the likelihood (Equation (4)). Analogously we can write

$$p^* = \arg\min_p \underbrace{-\pi(k^{\mathrm{exp}}|p)}_{=\mathrm{disc}(k(p),k^{\mathrm{exp}})}, \tag{B.5}$$

and define a discrepancy function. The maximum posterior estimate can be found similarly by defining

$$\mathrm{disc}(k(p), k^{\mathrm{exp}}) = -\pi(p|k^{\mathrm{exp}}) \propto -\pi(k^{\mathrm{exp}}|p)\pi(p), \tag{B.6}$$

where $\pi(p|k^{\mathrm{exp}})$ is the pdf of the posterior, which is proportional to the product of the likelihood pdf and the prior pdf $\pi(p)$.

In the special case of a Gaussian likelihood and a Gaussian prior, the problem to find the maximum posterior can be written as

$$p^* = \arg\min_p ||k(p) - k^{\text{exp}}||^2_{\Sigma_L^{-1}} + ||p - \mu_p||^2_{\Sigma_P^{-1}}, \tag{B.7}$$

where Σ_L and Σ_P are the covariance matrices of likelihood and prior and μ_P the mean of the prior. With $||x||^2_A = x^T Ax$ we denote weighted least squares. The special case of the maximum posterior approximation with Gaussian probabilities (Equation (B.7)) is also known as Tikhonov regularization, a well studied regularization technique [Benning and Burger, 2018]. When neglecting the regularization term $||p - \mu_p||^2_{\Sigma_P^{-1}}$ and choosing an isotropic covariance $\Sigma_L = \sigma_L^2 I$, Equation (B.7) is equivalent to minizing the squared error (Equation (B.3)).

A justification of the Gaussian assumptions or the definition of a proper likelihood and prior is beyond the scope of this work, because it requires a detailed uncertainty quantification of modeling errors and measurement noise. Furthermore, it is unclear if a general definition of *the objective function* for reconstruction in EPMA is possible, because the inverse problem is so multifaceted (c.f. Section 2). Therefore, we remain generic in the definition of the objective in this work.

B.1.1 Measuring Reconstruction Quality

Assume that for some objective function $f(p) = \text{disc}(k(p), k^{\text{exp}})$ the minimum p^* has been unveiled. (The process of unveiling p^* is the focus of Section B.1.2 et seq.) We might ask the question: How good is the reconstruction ?

Value of the Objective An obvious measure for the reconstruction quality is the value of the objective $f(p^*)$. The reconstruction result p^* was found by minimizing the value of f, so the smaller the objective, the better the model reproduces the measurements and the better p^* represents the actual material.

In many simple inverse problems, the value of the objective is an unambiguous quality measure for the result. However, consider the case where a model cannot represent certain features which are present in the measurements, but can to some extent counteract for the missing features by over- or underestimation in its parameters. The value of objective f would be smaller, but the reconstruction result would be worse.

In machine learning this effect is called underfitting [Goodfellow et al., 2016] and several strategies to detect underfitting are investigated. Also the opposite effect, overfitting, where a model is able to represent too many features of the measurements has to be taken care of.

Additional Measurements The disadvantages of using the value of the objective function as a reconstruction quality measure arise from the fact, that p^* is computed by minimizing the value of the objective function. By utilizing additional measurements k_+^{exp}, which are not included in the discrepancy $\text{disc}(k(p), k^{\text{exp}})$ used find p^*, the discrepancy of the model to the additional measurements does not suffer from over- or underfitting. Hence, a quality measure which is applicable in a practical reconstruction problem, is

$$\text{disc}(k_+(p^*), k_+^{\text{exp}}). \tag{B.8}$$

An analog to the additional measurements $k_+^{\exp}$ in machine learning is the test data set and the error (Equation (B.8)) is usually referred to as the test error.

Material Error A natural measure for the reconstruction quality is

$$|| \rho(x; p^*) - \rho^{\text{true}}(x)||_{L^2(\mathfrak{S})}, \tag{B.9}$$

where $\rho(x; p^*)$ are mass concentrations of the reconstructed material and ρ^{true} are mass concentrations of the true material. We interpret the mass concentrations as a general way to describe materials (c.f. Section A.1.1), and also interpret their difference (in the sense of the L^2-norm) as the difference of two materials. The material error is not suitable for a real reconstruction problem, because the true mass concentrations $\rho^{\text{true}}(x)$ have to be known. We can however utilize the material error in synthetic reconstruction problems, where the measurements are simulated based on a known material.

Parametrization Constraints Recall the constraints defined for the mass concentration models in Section A.3.1.4. If we derive mass concentrations based on one of those models, the underlying quantity ω_i, φ_i or γ_i has to fulfill the constraint to add up to 1. If we do not strongly impose this constraint for the reconstruction result p^*, its deviation from 1 can be thought of a reconstruction quality measure

$$||1 - \sum_{i=1}^{n_e} \{\omega, \varphi, \gamma\}_i(x)||_{L^2(\mathfrak{S})}. \tag{B.10}$$

Note that this is also similarly applied for classical reconstruction techniques in EPMA.

B.1.2 Optimization Methods

Having defined the reconstruction problem as an optimization problem, in this section we discuss iterative gradient-based methods, which solve optimization problems. With $p \in \mathbb{R}^n$ we denote the vector of unknowns and with $f : \mathbb{R}^n \to \mathbb{R}$ the objective function. Let f be twice differentiable with respect to the parameters p with the notation of ∇f for its gradient and Hf for its Hessian.

All optimization methods considered in this section are of the following iterative form: Given an initial guess of the parameters p_0 we update the current minimizer p_i according to

$$p_{i+1} = p_i + \Delta p_i, \tag{B.11}$$

until a convergence criterion is achieved (e.g. the objective $f(p_i)$ stays almost constant or the gradient $\nabla f(p_i)$ is sufficiently small).

We derive the optimization methods from the Taylor expansion of $f(p)$ around $q \in \mathbb{R}^n$

$$f(p) = f(q) + \nabla f(q)^T (p - q) + \frac{1}{2}(p - q)^T Hf(q)(p - q) + O(||p - q||^3) \tag{B.12}$$

Inserting the update of the optimization routines (Equation (B.11)) into the Taylor expansion, by replacing $p = p_{i+1}$ and $q = p_i$ yields

$$f(p_{i+1}) = \underbrace{f(p_i) + \nabla f(p_i)^T \Delta p_i + \frac{1}{2}\Delta p_i^T Hf(p_i)\Delta p_i}_{:= T_{p_i}^2(\Delta p_i)} + O(||\Delta p_i||^3). \tag{B.13}$$

Thereby, $T^2_{p_i}$ is the second order Taylor polynomial of f around the current minimizer p_i. The approximation $f(p_{i+1}) = T^2_{p_i}(\Delta p_i)$ is accurate, if the distance between p_{i+1} and p_i is sufficiently small, $||\Delta p_i||^3 \ll 1$.

Gradient Descent Method The gradient descent method bases on two simple steps:

- find the (unit) direction d_i of steepest descent ($||d_i|| = 1$);
- find a suitable step length $\alpha_i > 0$, such that the function decreases.

Then the update is given by

$$\Delta p_i = \alpha_i d_i. \tag{B.14}$$

To derive the direction of steepest descent d_i, we insert the update Δp_i into the first order Taylor expansion of f

$$f(p_{i+1}) \approx f(p_i) + \alpha_i \nabla f(p_i)^T d_i. \tag{B.15}$$

Since $\alpha_i > 0$, we seek $d_i = \arg\min_{d^* \in \mathbb{R}^n} \nabla f(p_i)^T d^*$ where d^* is a unit direction $||d^*|| = 1$. In Nocedal and Wright [2006] the minimum is given by $d_i = \frac{-\nabla f(p_i)}{||\nabla f(p_i)||}$, the direction which points opposite to the gradient $\nabla f(p_i)$.

Generally, the first order Taylor expansion only holds in a small neighborhood around the value p_i (if $||\Delta p_i||^2 \ll 2$), so to guarantee a decrease of the objective f, the step length α_i has to be chosen properly. A decreasing step can be found using line search methods [Nocedal and Wright, 2006], which guarantee that $f(p_{i+1}) < f(p_i)$ and additionally guarantee that the step size α_i is not too small.

The gradient descent method is straightforward to implement, but has disadvantages. If the objective function has a strong curvature with respect to some parameters p and a weak curvature with respect to other parameters, the convergence of the method degrades. Because all gradients are scaled by α_i the method will either converge very slowly for parameters of weak curvature or overshoot parameters of strong curvature. In Nocedal and Wright [2006] this characteristic of the steepest descent method is quantified by the condition number of the Hessian $\kappa(Hf) = \frac{\lambda_{\max}(Hf)}{\lambda_{\min}(Hf)}$ (a measure for the different curvatures). If $\kappa \approx 1$ the function has similar curvatures with respect to all parameters, if $\kappa \gg 1$ the curvatures strongly vary. Like many inverse problems, we expect that our reconstruction problem is more sensitive to certain parameters (close to the beam surface) than to others. A fact that hinders the application of the gradient descent method for reconstruction in EPMA.

Momentum Method The idea of the momentum method can be summarized as follows: we choose α_i such that parameters with average curvature converge, but together with the gradient, we also add the previous update Δp_{i-1} to the current update Δp_i. Then, updates for parameters where the curvature is weak, grow over the iterations, while updates for parameters where the curvature is strong cancel with each iteration. Often the method is motivated using a ball rolling down a hill: the ball is accelerated by a force into the direction of steepest descent: $-\nabla f(x_i)$, but already has a certain momentum Δp_{i-1}.

The update for the momentum method is given by

$$\Delta p_i = -\alpha_i \nabla f(p_i) + \beta_i \Delta p_{i-1} \tag{B.16}$$

where α_i and β_i are hyperparameter defining the contributions of the gradient and the previous step to the current update.

A special case of the momentum method with precisely defined parameters α_i and β_i is the conjugate gradient method [Nocedal and Wright, 2006].

ADAM Recent developments in machine learning lead to the development of modern optimization methods [Kingma and Ba, 2017]. A method, which is successfully applied in many machine learning problems is ADAM (adaptive moment estimation), a method for stochastic optimization. Since machine learning models often employ huge data sets, the objective f is calculated using only a randomly chosen subset of the data for computational feasibility reasons. Due to the random selection of data, the objective becomes a random variable and the goal is to minimize the expected value $\mathbb{E}(f(p))$. The optimization method ADAM approximates the low-order statistical moments of $\mathbb{E}(f(p))$ using running averages m_i and v_i, which decay (with rates β_1 and β_2) over the iterations of the optimization.

In Kingma and Ba [2017] the method is defined by

$$\Delta p_i = -\alpha \frac{\hat{m}_i}{\sqrt{\hat{v}_i} + \epsilon}, \tag{B.17}$$

where the moments $\hat{m}_i$ and $\hat{v}_i$ are given by

$$\hat{m}_i = \frac{m_i}{1-\beta_1^i} \qquad \hat{v}_i = \frac{v_i}{1-\beta_2^i} \tag{B.18a}$$

$$m_t = \beta_1 m_{i-1} + (1-\beta_1)\nabla f(p_i) \qquad v_t = \beta_2 v_{i-1} + (1-\beta_2)\nabla f(p_i)^2. \tag{B.18b}$$

Thereby $\nabla f(p_i)^2$ is the elementwise square of the gradient. Initial guesses for the hyperparameters of the method β_1, β_2, α and ϵ are given in Kingma and Ba [2017]. The step length α can be derived from prior knowledge about possible ranges of the parameters.

Newton's method The gradient descent method, the momentum method and ADAM only require the gradient $\nabla f(p_i)$ of the objective function. In contrast, Newton's method is based on the second order Taylor polynomial $T^2_{p_i}$ and additionally requires the Hessian $Hf(p_i)$. The update for Newton's method is derived by minimization of $T^2_{p_i}$, respectively by finding Δp_i such that $\nabla T^2_{p_i}(\Delta p_i) = 0$. We can derive

$$Hf(p_i)\Delta p_i = -\nabla f(p_i). \tag{B.19}$$

If the Hessian $Hf(p_i)$ is positive definite, the Newton step is given by

$$\Delta p_i = -Hf(p_i)^{-1}\nabla f(p_i). \tag{B.20}$$

For a convex quadratic function, Newton's method requires only one step to unveil the optimum, because $f(p_{i+1}) = T^2_{p_i}(\Delta p_i)$. For general functions Newton's method is only guaranteed to converge, if the estimate p_i is already in vicinity to the optimum, and the second order approximation is sufficient.

However, the naive application of Newton's method to a general objective f has disadvantages:

- If the Hessian is not invertible, the direction Δp_i can not be computed.
- If the Hessian is not positive definite, the direction Δp_i must not be a descent direction.

- The computation of the Hessian $Hf(p_i)$ is expensive.

To overcome these disadvantages, Quasi-Newton methods have been developed. Instead of using the exact Hessian $Hf(p_i)$, it is approximated from previously computed objective values $f(p_i)$ and gradients $\nabla f(p_i)$. In Quasi-Newton methods the update Δp_i is

$$\Delta p_i = -B_i^{-1} \nabla f(p_i) \tag{B.21}$$

where B_i is an approximation of the Hessian at the i-th step.

BFGS In the BFGS method [Nocedal and Wright, 2006] the update is defined as

$$\Delta p_i = -\alpha_i B_i^{-1} \nabla f(p_i) \tag{B.22}$$

where B_i is the approximation of the Hessian. B_i is computed based on previous evaluations of the gradient $\nabla f(p_i)$ and carefully defined, such that it satisfies conditions, e.g. that Δp_i is a descent direction for the objective f. In Nocedal and Wright [2006] B_k is defined as

$$B_{i+1} = B_i - \frac{B_i s_i s_i^T B_i}{s_i^T B_i s_i} + \frac{y_i y_i^T}{y_i s_i} \tag{B.23}$$

where $s_i = p_{i+1} - p_i$ and $y_i = \nabla f(p_{i+1}) - \nabla\ f(p_i)$. However, Equation (B.23) is not intended to be implemented in this form, because it implicitly depends on the new minimizer p_{i+1}. For an implementation of BFGS and a variant (L-BFGS) which is less memory consuming we refer to Nocedal and Wright [2006]; Mogensen and Riseth [2018].

Common Optimizer Interface All presented iterative optimization methods rely on the computation of the objective function $f(\cdot)$ and the computation of the gradient $\nabla f(\cdot)$. There exist multiple packages in julia (e.g. `Optim.jl` [Mogensen and Riseth, 2018] or `Flux.jl` [Innes, 2018b]) which implement optimization routines, all of which share the requirement to provide implementations of f and its gradient ∇f. The computation of the objective f is covered by Chapter A, hence for the remainder of this thesis in Chapter B, we will focus on the systematic and generic computation of the gradient ∇f.

B.2 Algorithmic Differentiation and the Adjoint State Method

B.2.1 Why Algorithmic Differentiation

In Section B.1.2 the application of gradient-based optimization methods to solve the minimization problem is motivated. All methods require the evaluation of gradients $\nabla f(p)$. While for simple analytical functions, the derivation of the gradient is straightforward, the computation of the gradient for complex functions, e.g. the objective function used for the minimization, is challenging. Special attention must be paid to the computation time, which quickly grows to unmanageable orders of magnitude.

A naive approximation of the gradient can be computed using finite differences

$$(\nabla f(p))_i = \frac{f(p + he_i) - f(p)}{h}, \tag{B.24}$$

by successive perturbation of each parameter and approximation of the derivative by the secant of the function $f(p)$. But the application of finite differences involves multiple evaluations of the forward function $f(p)$ and therefore becomes computationally very expensive if the number of parameters n_p is large. Furthermore, it is unclear how to properly choose the step length h. An adaptive search for the step length, or the approximation of the derivative using higher order finite differences increases the number of function evaluations even more.

Consider the case of a reconstruction problem with a material parametrization with n_p parameters. The objective function combines the discrepancy of k-ratios measured using $n_{\mathcal{U}}$ beam energies and positions. Then, even the computation of the forward problem is expensive because the approximations of $n_{\mathcal{U}}$ solutions of the P_N-model are necessary. To approximate the gradient of the objective function at least n_p+1 computations of the forward problem are required, hence the solution of $n_{\mathcal{U}}(n_p+1)$ solutions of the P_N-model. Because of the scaling with the number of parameters, a finite difference approximation of the gradient is expensive.

Remedy for the problem of large computational cost is the application of adjoint methods for the computation of the gradient. Adjoint methods scale with the number of outputs of the considered function rather than with the number of inputs. For the objective function used on optimization $f(p) : \mathbb{R}^{n_p} \to \mathbb{R}$, this advantage is of specific interest, because the output is a scalar. In theory, the cost to compute the gradient of an objective function ∇f using adjoint methods is twice the cost of the computation f.

We distinguish between two kinds of adjoint methods, *adjoint algorithmic differentiation* and *the adjoint state method*. The former being a mode of algorithmic differentiation, a concept from computer science to compute derivatives of arbitrary program code. The latter is a method from optimal control theory to compute derivatives of functionals constrained by partial differential equations.

Algorithmic differentiation (AD) proposes to automatically derive derivatives for arbitrary program code. While the efficient application of AD to complex numerical codes is challenging, it can conveniently be applied to smaller functions, which are called during the execution of a complex numerical code. On the other hand, the adjoint state method is applied to compute derivatives of functionals based on partial differential equation solvers.

However, the naive implementation of the adjoint state method is specific to the objective function and the material parametrization and does not allow a convenient replacement of either.

We describe a combination of *adjoint algorithmic differentiation* and *the adjoint state method* that allows the convenient extension and exchange of the objective function and the material parametrization. Thereby we rely on the concepts of AD, but avoid the automatic application of AD to the full pde solver. The differentiation of additional material parametrizations or objective functions can either be handled by an automatic AD tool, if the generated adjoint version is performant enough, or by the definition of an additional adjoint implementation of the respective parametrization or objective. The core of the implementation remains unaffected.

B.2.2 Algorithmic Differentiation

Algorithmic differentiation [Naumann, 2011; Griewank, 2003] is a systematic way to compute exact derivatives of numerical program code. There exist two fundamental modes of AD, the tangent mode and the adjoint mode, both of which base on the chain rule of differentiation. We motivate tangent and adjoint mode AD based on the chain rule and mention their differences. Afterwards we precisely define both modes using single assignment code, discuss their implementation, justify our choice for the adjoint mode and demonstrate implementation examples.

Differentiation Chain Rule Given a (differentiable) function f, which is the composition of two other functions g and h

$$f = h \circ g, g : \mathbb{R} \to \mathbb{R}, h : \mathbb{R} \to \mathbb{R}, x \to h(g(x)). \tag{B.25}$$

Using the chain rule we can compute the derivative $\frac{\partial y}{\partial x}$ as

$$\frac{\partial y}{\partial x} = h'(g(x))g'(x). \tag{B.26}$$

Program code can be viewed as the (arbitrarily complex) composition of simple functions. Hence, the derivative of an output value can be viewed as the (arbitrarily long) product of derivatives of simple functions. The two modes of AD, tangent and adjoint, differ in the way in which they compute the product of derivatives. The product is commutative, hence it does not matter, whether we start multiplication at the end or in the beginning of the product. While the tangent mode starts with inputs of the numerical code and computes derivatives alongside the primal function evaluation, the adjoint mode begins at the outputs (after the primal code is evaluated) and computes the derivative backwards. Thereby the two modes differ in computational complexity and memory requirements. While for the accumulation of a Jacobian of a function $f : \mathbb{R}^n \to \mathbb{R}^m$ the tangent mode scales with the number of inputs n, the adjoint mode scales with the number of outputs m.

An efficient implementation of AD is challenging. While the tangent mode can be implemented memory efficient (because it can be computed alongside the primal function) the adjoint mode requires the storage of intermediate values. This quickly becomes too memory-consuming and strategies to reduce the memory requirements are necessary.

B.2.2.1 Single Assignment Code

For the definition of tangent and adjoint mode, consider a function $f : \mathbb{R}^n \to \mathbb{R}^m, x \to y$, which is split up into individual single assignments φ_j by the introduction of scalar intermediate variables $v_j \in \mathbb{R}$.

$$(v_1, \ldots, v_n)^T = x \tag{B.27a}$$

$$v_j = \varphi_j((v_l)_{l \prec j}) \tag{B.27b}$$

$$y = (v_{|V|-m}, \ldots, v_{|V|})^T, \tag{B.27c}$$

Using tangents $\dot{v}_j \in \mathbb{R}$ and adjoints $\bar{v}_j \in \mathbb{R}$ of the intermediate variables v_j we can define the two modes of AD as

$$\dot{v}_j = \sum_{k \prec j} \frac{\partial \varphi_j((v_l)_{l \prec j})}{\partial v_k} \dot{v}_k \tag{B.28a}$$

$$\bar{v}_k = \sum_{j \succ k} \frac{\partial \varphi_j((v_l)_{l \prec j})}{\partial v_k} \bar{v}_j. \tag{B.28b}$$

For the tangent $\dot{v}_j$ of the intermediate variable v_j all tangents $\dot{v}_k$ which precede $k \prec j$ (all v_k which on which v_j directly depends on) are accumulated, multiplied by the respective partial derivatives. For the adjoint $\bar{v}_k$ of the intermediate variable v_k all adjoints $\bar{v}_k$ which succeed $j \succ k$ (all v_j which directly depend on v_k) are accumulated, multiplied by the respective partial derivatives. To compute the partial derivative for both modes the values of the primal intermediate variables $(v_l)_{l \prec j}$ are necessary.

A convenient way to visualize AD is the notion of a directed acyclic graph (DAG, also computational graph). Every intermediate variable v_j and its assignment function φ_j is associated with a node and every partial derivative is associated with an edge. Then the tangent $\dot{v}_j$ of a node v_j is the sum over all incoming edges, while the adjoint $\bar{v}_j$ of a node is the sum over all outgoing edges.

An exemplary graph is drawn in Figure B.1. Thereby the input is $(v_1, v_2)^T = x$, the output is $y = (v_5, v_6)^T$ and v_3 and v_4 are intermediate variables. We visualize the edges (the partial derivatives) which are necessary for the accumulation of the tangent $\dot{v}_3 = \frac{\partial \varphi_3}{\partial v_1}\dot{v}_1 + \frac{\partial \varphi_3}{\partial v_2}\dot{v}_2$ by dotted arrows and the edges which are necessary for the accumulation of the adjoint $\bar{v}_4 = \frac{\partial \varphi_6}{\partial v_4}\bar{v}_6 + \frac{\partial \varphi_5}{\partial v_4}\bar{v}_5$ by dashed arrows.

Short inspection unveils the fact, that the initialization/seeding of tangents of input variables and the initialization/seeding of adjoints of output variables is necessary. Seeding a unit vector to the tangents of the inputs, the tangents of the outputs hold the respective column of the Jacobian of f. Analogously, seeding a unit vector to the adjoints of the output, the adjoint of the inputs hold the respective row of the Jacobian. However, if f is only a part of a larger computation, the tangents/adjoints of the inputs/outputs can also be specified from the preceding/succeeding part of the computation.

The disassembly of large numerical functions into single assignments is academic, and it is often sufficient to analyze functions on a higher level. The definition of tangents and adjoints in Equation (B.28) can be extended to vector-valued intermediate variables v_j by

$$\dot{v}_j = \sum_{k \prec j} \left(\frac{\partial \varphi_j((v_l)_{l \prec j})}{\partial v_k} \right) \dot{v}_k \tag{B.29a}$$

$$\bar{v}_k = \sum_{j \succ k} \left(\frac{\partial \varphi_j((v_l)_{l \prec j})}{\partial v_k} \right)^T \bar{v}_j, \tag{B.29b}$$

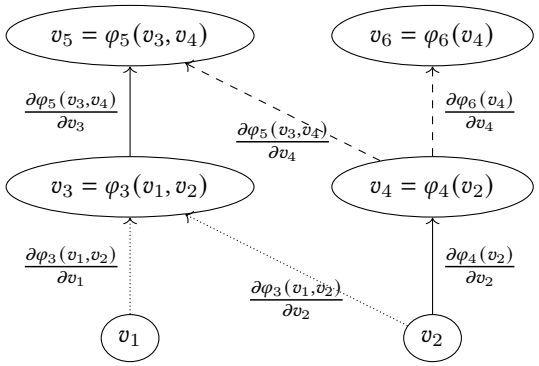


Figure B.1: *The directed acyclic graph (DAG) of a simple function with two inputs v_1 and v_2, two intermediate variables v_3 and v_4 and two outputs v_5 and v_6. Arrows denote direct dependencies of variables. The dotted arrows depict the tangent accumulation of $\dot{v}_3$, the dashed arrows the reversed accumulation of the adjoint $\bar{v}_4$.*

`Zygote.jl`	adjoint - source-to-source transformation	[Innes, 2018a]
`ReverseDiff.jl`	adjoint - operator overloading	[Kelley, 2021]
`ForwardDiff.jl`	tangent - operator overloading	[Revels et al., 2016]

Table B.1: *An excerpt of the rich world of implementations of AD in the programming language julia, annotated with name, AD mode and implementation model. We mention the libraries that were worked with mostly in the course of this work.*

where $\left(\frac{\partial \varphi_j}{\partial v_k}\right) \cdot$ is the Jacobian-(for adjoints transposed Jacobian)-vector product.

Equation (B.28) and Equation (B.29) sum up the fundamental functionality of AD; however, the efficient application is challenging and a field of open research [Naumann, 2011].

B.2.2.2 Implementations of Algorithmic Differentiation

The two modes of AD are usually implemented either by operator overloading or by source-to-source transformation. In the former approach, operator-overloading, a dual number type (the primal function value with its tangent or adjoint) is defined, and all primal functions are overloaded such that they additionally perform the functionality required for tangent or adjoint mode AD. For tangent mode that is the accumulation of tangents in the dual of the output value, for adjoint mode the operation is usually recorded to a tape, which is reversed for the adjoint accumulation.

The latter approach is a source-to-source transformation, where tangent and adjoint functions are generated and implemented alongside the primal function code. This offers the possibility of applying subsequent compiler optimizations. Table B.1 lists some implementations of AD in the programming language julia that were worked with in the course of this work. Mostly we rely on `Zygote.jl` [Innes, 2018a] a source-to-source adjoint AD tool, which we enhance by the custom definition of adjoint versions of our implemented functions.

We mention some technicalities of implementing adjoint mode AD, which we rely on later:

Accumulation of Adjoints If the primal code contains the subsequent execution of two function (g and h) which depend on the same variable v, the adjoints are accumulated.

$$\begin{aligned} y &= f(v) \\ z &= g(v), \end{aligned} \tag{B.30a}$$

$$\begin{aligned} \bar{v} &= (\frac{\partial g(v)}{\partial v})^T \bar{z} \\ \bar{v} &\mathrel{+}= (\frac{\partial f(v)}{\partial v})^T \bar{y} \end{aligned} \tag{B.30b}$$

Linear Functions If the function to be differentiated is linear in an argument, its partial derivative with respect to that argument is constant, and the adjoint code does not have to memorize the primal function value to determine the derivative.

Checkpointing Adjoint AD has the disadvantage of having a high memory requirement. The concept of checkpointing reduces the memory requirements by sacrificing a longer runtime. Instead of storing all intermediate values during the forward pass, only the input values of subfunctions are memorized. During the adjoint pass, the intermediate values have to be recomputed from the stored input values. Checkpointing subsequently repeats the forward pass for subfunctions and reverses them iteratively.

B.2.3 Examples of Adjoint Mode AD

B.2.3.1 Adjoint of the Mass Fraction Model in Zygote

Recall the relation between mass concentrations ρ and mass fractions ω in Equation (A.65). We might implement the equation using an intermediate variable R (Equation (B.31a)). Using the partial derivatives $\frac{\partial \rho_i}{\partial R} = -R^{-2}\omega_i$, $\frac{\partial \rho_i}{\partial \omega_i} = R^{-1}$ and $\frac{\partial R}{\partial \omega_i} = \frac{1}{\rho_i^{\text{pure}}}$ we can derive the adjoint version (Equation (B.31b)) of the relation.

$$\begin{aligned} R &= \sum_{j=1}^{n_e} \frac{\omega_j}{\rho_j^{\text{pure}}} \\ \rho_i &= \omega_i R^{-1}. \end{aligned} \tag{B.31a}$$

$$\begin{aligned} \bar{R} &= -R^{-2} \sum_{i=1}^{n_e} \omega_i \bar{\rho}_i \\ \bar{\omega}_i &= R^{-1} \bar{\rho}_i \\ \bar{\omega}_i &\mathrel{+}= \frac{1}{\rho_i^{\text{pure}}} \bar{R} \end{aligned} \tag{B.31b}$$

The computation of the adjoint $\bar{R}$ accumulates all adjoints from $\bar{\rho}_i$, because during the forward pass the same R was used to compute every ρ_i. Due to the nonlinearity in R, the value of R is required for the partial derivative. In the forward pass, the mass fraction ω_i is used for the computation of R and ρ_i, hence we accumulate its adjoints using +=. Note that the order of the operations in the adjoint version is reversed regarding the primal function evaluation, because adjoints of subsequent variables are required to compute adjoints of the preceding variables.

An implementation of the forward Equation (B.31a) and adjoint Equation (B.31b) is presented in Figure B.2 as a custom adjoint implementation following the julia library `Zygote.jl`. If called outside the context of Zygote, the function `mass_concentrations(`ω`)` only returns ρ. Otherwise, it additionally returns the adjoint function `mass_concentrations_adjoint(`ρ`_adjoint)`, a closure (a function with enclosed data: ω, ρ`_pure` and `R`)

```
@adjoint function mass_concentrations(ω)
    R = sum(ω ./ ρ_pure)
    ρ = ω ./ R
    function mass_concentrations_adjoint(ρ_adjoint)
        R_adjoint = -dot(ω, ρ_adjoint)/R^2
        ω_adjoint = ρ_adjoint ./ R
        ω_adjoint += R_adjoint ./ ρ_pure
        return ω_adjoint
    end
    return ρ, mass_concentrations_adjoint
end
```

Figure B.2: *An implementation of the adjoint of the mass concentration model in the form of custom adjoint definitions used by* `Zygote.jl`*. Given mass fractions* ω_i *the forward pass computes mass concentrations* ρ *as described in Equation* (B.31a)*. The adjoint function defines a closure which encapsulates* ω *and* `R` *from the forward pass and computes the respective adjoints of the mass fractions* `ω_adjoint` *as described in Equation* (B.31b)*.*

which for given adjoints of the output `ρ_adjoint`, computes the adjoints of the inputs `ω_adjoint`. After the evaluation of a forward pass, that includes a call to `mass_concentrations(ω)`, Zygote reverses the forward pass and meanwhile calls our definition of the adjoint function `mass_concentrations_adjoint(ρ_adjoint)`.

To apply checkpointing for this example, the adjoint closure `mass_concentrations_adjoint` of the function would not enclose the intermediate variable `R`, but recompute `R = sum(ω ./ ρ_pure)` prior to the computation of `R_adjoint`. Checkpointing would probably not have a noticeable effect on the runtime here, but this example demonstrates the technique which is applied in our implementation.

B.2.3.2 Adjoint of Weighted Least Squares with Regularization

Recall the equation for weighted least squares (Equation (B.7)). Here we derive the adjoint version of a function which has the form Equation (B.32a) where for simplicity, we replaced the residuum $k(p) - k^{\text{exp}}$ with e, the normalization distance $p - \mu_p$ with q and the inverse covariance matrices with A and B.

$$f = e^T A e + q^T B q, \tag{B.32a}$$

$$\begin{aligned} \bar{e} &= (A + A^T) e \bar{f} \\ \bar{A} &= e e^T \bar{f} \\ \bar{q} &= (B + B^T) q \bar{f} \\ \bar{B} &= q q^T \bar{f} \end{aligned} \tag{B.32b}$$

The adjoint version (Equation (B.32b)) can be derived from the partial derivatives $\frac{\partial f}{\partial \cdot}$. If one of the matrices A or B is symmetric we can replace e.g. $(A + A^T)$ by $2A$ and derive the partial derivative $\frac{\partial e^T A e}{\partial e} = 2Ae$.

Using Equation (B.32a) as the weighted least squares objective function, the adjoints of the matrices A and B are usually not required, because the covariance matrices do not depend on the optimized model parameters. Only e and q depend on the model parameters and the adjoints $\bar{e}$ and $\bar{q}$ are required. We included the adjoints $\bar{A}$ and $\bar{B}$ for the sake of a complete illustration.

B.2.3.3 Adjoint of the Neural Network Parametrization

The adjoint version of the neural network parametrization, which we introduced in Section A.3.3.3, is the computation of derivatives as part of the Backpropagation algorithm, a method frequently applied in machine learning [Goodfellow et al., 2016; Bishop, 2006]. Here we only denote the adjoint version of the computation of a dense layer, where the forward computation can be implemented by Equation (B.33a).

$$\begin{aligned} y^k &= W^k x^k + b^k \\ z^k &= \sigma^k(y_k) \end{aligned} \tag{B.33a}$$

$$\begin{aligned} \bar{y}^k &= \left(\frac{\partial \sigma_k(y_k)}{\partial y_k}\right)^T * \bar{z}^k \\ \bar{W}^k &= \bar{y}^k \left(x^k\right)^T \\ \bar{x}_k &= \left(W^k\right)^T \bar{y}^k \\ \bar{b}_k &= \bar{y}^k \end{aligned} \tag{B.33b}$$

In its adjoint version (Equation (B.33b)), the $*$ denotes elementwise multiplication, other products are matrix multiplications. The computation of derivatives of the whole network is implemented by chaining the adjoint versions of each layer in reversed order.

B.2.4 The Adjoint State Method

Unlike algorithmic differentiation on the numerical level, the adjoint state method [Plessix, 2006] offers a way to derive derivatives of functions that include implicit relations of variables. Bünger [2021]; Claus et al. [2021] already successfully apply the adjoint state equation to compute gradients in the context of EPMA. In Ma et al. [2021] combinations of algorithmic differentiation and the adjoint state method for time stepping schemes (mainly in the context of ODEs) are discussed. Here we only present the steps of the derivation of the adjoint state method, which are required to apply it to the P_N-model and derive the adjoint formulation of the P_N operator. For a detailed derivation of the adjoint state method we refer to the mentioned literature. In the next section (Section B.2.6) we discuss our implementation which combines the adjoint state method with algorithmic differentiation.

Consider a function $f : \mathbb{S} \to \mathbb{R}$ where its input $\psi \in \mathbb{S}$ is implicitly defined by $g(\rho, \psi) = 0$ with $\rho \in \mathbb{R}^n$. In this section the variables ρ and ψ are defined in a general way, but foreshadow their counterpart in the P_N-model. We define the output of the function $f(\psi)$ as $y \in \mathbb{R}$

$$y = f(\psi) \quad | \quad g(\rho, \psi) = 0, \tag{B.34}$$

and seek the derivative of y with respect to ρ. The adjoint state method to compute the derivative $\frac{\partial y}{\partial \rho}$ consists of three main steps:

- The computation of a realization $\hat{\psi}$ of the implicit constraint $g(\rho, \hat{\psi}) = 0$, given ρ.
- The computation of the adjoint state variable λ using the adjoint state equation

$$\frac{\partial f}{\partial \psi} + \left[\frac{\partial g}{\partial \psi}\right]^* \lambda = 0, \tag{B.35}$$

  where $\left[\frac{\partial g}{\partial \psi}\right]^*$ is the adjoint of the derivative operator $\left[\frac{\partial g}{\partial \psi}\right]$. The adjoint operator is defined as

$$\langle \lambda, \left[\frac{\partial g}{\partial \psi}\right] \delta\psi \rangle = \langle \left[\frac{\partial g}{\partial \psi}\right]^* \lambda, \delta\psi \rangle. \tag{B.36}$$

where $\delta\psi$ is a perturbation in ψ due to a perturbation $\delta\rho$ in ρ because of the relation $g(x + \delta x, \psi + \delta\psi) = 0$.

- And the computation of the gradient

$$\frac{\partial y}{\partial \rho} = \frac{\partial}{\partial \rho}\langle \lambda, g(\rho, \hat{\psi})\rangle. \tag{B.37}$$

The advantage of the adjoint state method is that it avoids the computation of $\frac{\partial \hat{\psi}}{\partial \rho}$. The derivative of the realization $\hat{\psi}$ with respect to the variable ρ could be approximated using e.g. finite differences or sensitivity equations [Ma et al., 2021]. However, both methods scale with the number of input parameters which hinders application to our problem. Additionally, $\frac{\partial \hat{\psi}}{\partial \rho}$ is high-dimensional and its storage very memory-consuming. Avoiding the computation of $\frac{\partial \hat{\psi}}{\partial \rho}$ saves long computing times and clearly motivates the use of the adjoint state method. The computational costs of the adjoint method are:

- The computation of the realization $\hat{\psi}$ is virtually free. To compute the value y of primal function $f(\psi)$, a realization of the constraint $\hat{\psi}$ is calculated anyway. The additional cost of the adjoint state method is the storing of $\hat{\psi}$, because it is required later.
- The adjoint state equation (Equation (B.35)) is often similarly complex as the primal equation. Only the adjoint source $\frac{\partial f}{\partial \psi}$ must be added. Then the cost of solving the adjoint state equation is comparable to the cost of solving the primal constraint $g(\rho, \psi)$.
- The cost to compute Equation (B.37) depends on the relation of the constraint $g(\rho, \psi)$ and ρ. If the relation is linear (as in our case c.f. Section B.2.6) Equation (B.37) can be computed very efficiently.

Compared to finite differences or sensitivity equations, the main advantage of the adjoint method is, that it does not scale in the number of inputs (the number of variables in $\rho \in \mathbb{R}^n$). In case $f(\psi)$ is multivariate, the number of evaluations of Equation (B.35) would scale in the number of outputs, if $f(\psi)$ is a scalar, one solution of the adjoint state equation is sufficient to compute the whole gradient.

B.2.4.1 Adjoint Operator of the PN-model

The implicit relation (c.f. $g(\rho, \psi) = 0$ in Equation (B.34)) for the P_N model is the governing partial differential equation (Equation (A.21)). To derive the adjoint state equation of the P_N model, first the adjoint operator $\left[\frac{\partial g}{\partial \psi}\right]^*$ must be deduced. Due to the linearity of the partial differential equation $g(\rho, \psi)$ in the moments ψ, we can write $\left[\frac{\partial g}{\partial \psi}\right]\delta\psi = g(\rho, \delta\psi)$.

We derive the adjoint operator of $\left[\frac{\partial g}{\partial \psi}\right]\delta\psi$ using its definition in Equation (B.36). The scalar product is the inner product in the function space L^2, therefore given by integration and multiplication $\langle \cdot, \cdot \rangle = \int_{\epsilon_{\text{cut}}}^{\epsilon_{\text{init}}} \int_{\mathfrak{S}} \cdot^T \cdot \, \mathrm{d}x \, \mathrm{d}\epsilon$. Then we can derive the adjoint operator using integration by parts. Additionally, we exploit the symmetry of the advection matrices $A^{(n)} =$

$A^{(n)T}$.

$$
\begin{aligned}
&\langle \lambda, \frac{\partial g}{\partial \psi}\delta\psi\rangle \\
&\quad= \int_{\epsilon_{\text{cut}}}^{\epsilon_{\text{init}}} \int_{\mathfrak{S}} \lambda^T(-\partial_\epsilon(S\delta\psi) + A^{(x)}\partial_x\delta\psi + A^{(y)}\partial_y\delta\psi + A^{(z)}\partial_z\delta\psi + C\delta\psi)\,\mathrm{d}x\,\mathrm{d}\epsilon \\
&\quad= \int_{\epsilon_{\text{cut}}}^{\epsilon_{\text{init}}} \int_{\mathfrak{S}} \underbrace{(S\partial_\epsilon\lambda - A^{(x)}\partial_x\lambda - A^{(y)}\partial_y\lambda - A^{(z)}\partial_z\lambda + C\lambda)^T}_{:=\left[\frac{g}{\psi}\right]^*\lambda}\delta\psi\,\mathrm{d}x\,\mathrm{d}\epsilon \\
&\qquad+ \underbrace{\int_{\epsilon_{\text{cut}}}^{\epsilon_{\text{init}}} \int_{\mathfrak{S}} -\partial_\epsilon(S\lambda^T\delta\psi) + \partial_x(\lambda^T A^{(x)}\delta\psi) + \partial_y(\lambda^T A^{(y)}\delta\psi) + \partial_z(\lambda^T A^{(z)}\delta\psi)\,\mathrm{d}x\,\mathrm{d}\epsilon}_{:=0} \\
&\quad= \langle \left[\frac{\partial g}{\partial \psi}\right]^* \lambda, \delta\psi\rangle
\end{aligned}
\tag{B.38}
$$

Together with the adjoint source $\frac{\partial f}{\partial \psi}$, the adjoint operator $\left[\frac{\partial g}{\partial \psi}\right]^*\lambda$ form the partial differential equation that governs the adjoint state variable λ. The boundary integrals, which are set to zero in Equation (B.38) yield boundary conditions for the adjoint state variable λ. From the first term

$$
\left[\int_{\mathfrak{S}} -S\lambda^T\delta\psi\,\mathrm{d}x\right]_{\epsilon_{\text{cut}}}^{\epsilon_{\text{init}}} = 0, \tag{B.39}
$$

we derive an initial condition for the adjoint state variable λ. At ϵ_{init} the solution variable is specified by its initial condition $\psi(\cdot, \epsilon_{\text{init}}) = \psi_0(\cdot)$, hence no perturbation in the state variable is implied $\delta\psi(\epsilon_{\text{init}}) = 0$. For Equation (B.39) to hold, we must prescribe the adjoint state variable $\lambda(\cdot, \epsilon_{\text{cut}}) = 0$ at the energy ϵ_{cut}.

The remaining terms of the boundary integrals in Equation (B.38) require

$$
\int_{\epsilon_{\text{cut}}}^{\epsilon_{\text{init}}} \lambda^T A^{(n)}\delta\psi\,\mathrm{d}\epsilon = 0 \quad \forall x \in \partial\mathfrak{S}. \tag{B.40}
$$

From the boundary conditions of ψ in Equation (A.30), we derive a relation between odd and even perturbations at the spatial boundary $\delta\psi^{o,n} = L_o^{(n)}A_{o,e}^{(n)}\delta\psi^{e,n}$. Defining the boundary conditions of the adjoint state equation λ as $\lambda^{o,n} = -L_o^{(n)}A_{o,e}^{(n)}\lambda^{e,n}$ we find that Equation (B.40) holds.

$$
\begin{aligned}
\lambda^T A^{(n)}\delta\psi &= \begin{pmatrix}\lambda_{e,n}\\ \lambda_{o,n}\end{pmatrix}^T \begin{pmatrix}0 & A_{e,o}^{(n)}\\ A_{o,e}^{(n)} & 0\end{pmatrix}\begin{pmatrix}\delta\psi_{e,n}\\ \delta\psi_{o,n}\end{pmatrix} \\
&= \lambda_{e,n}^T A_{e,o}^{(n)}\delta\psi_{o,n} + \lambda_{o,n}^T A_{o,e}^{(n)}\delta\psi_{e,n} \\
&= \lambda_{e,n}^T A_{e,o}^{(n)}(L_o^{(n)}A_{o,e}^{(n)}\delta\psi_{e,n}) - (L_o^{(n)}A_{o,e}^{(n)}\lambda_{e,n})^T A_{o,e}^{(n)}\delta\psi_{e,n} \\
&= \lambda_{e,n}^T A_{e,o}^{(n)}(L_o^{(n)}A_{o,e}^{(n)}\delta\psi_{e,n}) - \lambda_{e,n}^T A_{o,e}^{(n)T}L_o^{(n)T}A_{o,e}^{(n)}\delta\psi_{e,n} = 0.
\end{aligned}
\tag{B.41}
$$

Thereby we used the fact, that $L_o^{(n)T} = L_o^{(n)}$ and that the blocks of $A_{e,o}^{(n)T} = A_{o,e}^{(n)}$ are symmetric. Symmetry can be verified using the definition of both matrices (Equation (A.27) and Equation (C.1)).

Combining the adjoint operator of the P_N-equation (Equation (B.38)) with the initial

and boundary conditions we arrive at the following problem for the adjoint state variable λ.

$$\begin{aligned} S\partial_\epsilon \lambda - A^{(x)}\partial_x \lambda - A^{(y)}\partial_y \lambda - A^{(z)}\partial_z \lambda + C\lambda &= -\frac{\partial f}{\partial \psi} \quad \forall \epsilon \in [\epsilon_{\text{cut}}, \epsilon_{\text{init}}], x \in \mathfrak{S} \\ \lambda(\epsilon = \epsilon_{\text{cut}}, x) &= 0 \quad \forall x \in \mathfrak{S} \\ \lambda^{o,n} &= -L_o^{(n)} A_{o,e}^{(n)} \lambda^{e,n} \quad \forall \epsilon \in [\epsilon_{\text{cut}}, \epsilon_{\text{init}}], x \in \partial\mathfrak{S} \end{aligned} \tag{B.42}$$

Equation (B.42) inherits many properties from the P_N-equation Equation (A.21). The matrices S, $A^{(n)}$ and C still posses the structural properties to be solved using StaRMAP. The differences to Equation (A.21) are:

- The characteristics regarding the energy ϵ are reversed. While the P_N-equation is solved "backwards" in energy, the adjoint state equation is solved "forwards" (from ϵ_{cut} to ϵ_{init}).
- The stopping power S appears outside the partial derivative ∂_ϵ, hence no special treatment is necessary.
- The right-hand side describes an additional source term, which can be handled by the source term Q defined for StaRMAP in Equation (A.33).

In analogy to the definition of the operator which computes one step of the P_N-equation in Equation (A.59), we define an operator which computes one step of the adjoint state equation (Equation (B.42)) using StaRMAP

$$\lambda_{i+1} = P_N^{*,i\to i+1}(\lambda_i, \rho, \frac{\partial f}{\partial \psi}). \tag{B.43}$$

Note that f is purposefully left arbitrary in this section. We will elaborate f and the source of the adjoint state equation $\frac{\partial f}{\partial \psi}$ in Section B.2.6. In the next section, we motivate the combination of the adjoint state method and adjoint mode algorithmic differentiation based on a simple example. Afterwards, in Section B.2.6 we extend the ideas presented for the simplified model to the full k-ratio model from Chapter A.

B.2.5 Motivation of the Differentiation Method based on a Simplified Forward Model

The concept to combine the adjoint mode of algorithmic differentiation and the adjoint state method is illustrated by a simple example based on a linear scalar ordinary differential equation (ODE). In the analogous derivation of the combined method for the k-ratio model in Section B.2.6, the actual purity of the method is overshadowed by technicalities due to the complexity of the k-ratio model.

We consider the following forward model based on a linear ODE. The solution variable is $\xi(t)$, which relates to some parameters p through a quantity $m(t, p)$ in the ODE. In analogy to our k-ratio model, we assume additivity of $m(t, p)$ using $\rho(p)$. The relation between $\rho(p)$

and the parameters p remains general.

$$\begin{aligned} &\overbrace{\partial_t \xi(t) - \underbrace{\left(m_1(t)\rho_1(p) + m_2(t)\rho_2(p)\right)}_{:=m(t,p)} \xi(t)}^{:=g(\xi,p)} = 0 \quad \xi(0) = \xi_0 \\ &I = \int_0^T h(t,p)\xi(t)\,\mathrm{d}t \end{aligned} \tag{B.44}$$

The output of the forward model is I, which is related to $\xi(t)$ by an integral relation with additional $h(t,p)$.

The desired generality in the relation $\rho(p)$ and $y(I)$, an imaginary scalar output variable, is achieved by the following observation: Consider the application of adjoint mode algorithmic differentiation to compute the derivative of y. Adjoint mode AD seeds the scalar adjoint of the output variable $\bar{y} = 1$, hence, after being completely computed, the adjoint of any other preceding variable v is

$$\bar{v} = \frac{\partial y}{\partial v} = \left(\frac{\partial w}{\partial v}\right)^T \bar{w}, \tag{B.45}$$

where w is another intermediate variable and $\bar{w}$ its adjoint. If the adjoint $\bar{w}$ is given, we do not have to consider a relation to the output y for the computation of the adjoint $\bar{v}$, only the relation between w and v is of interest. Hence, the consideration of the adjoint $\bar{w}$ allows generality in the relation between w and y.

Thus, for the further analysis of the simplified forward model (Equation (B.44)) the encapsulation between given adjoints $\bar{I}$ and computed adjoints $\bar{\rho}_1$ and $\bar{\rho}_2$ is sufficient. Using Equation (B.45) the two connections between the adjoint state method and adjoint mode algorithmic differentiation become apparent. The source in the adjoint state equation (Equation (B.35)) is

$$\frac{\partial f}{\partial \psi} = \frac{\partial f}{\partial I}\frac{\partial I}{\partial \psi} = \bar{I} h(t,p), \tag{B.46}$$

using the Fréchet derivative $\frac{\partial I}{\partial \psi} = h(t,p)$. Additionally, we identify the gradient in the adjoint state method (Equation (B.37)) with

$$\bar{\rho}_i = \frac{\partial y}{\partial \rho_i}. \tag{B.47}$$

Then for the simplified forward model in Equation (B.44) we can derive the following adjoint relation.

$$\begin{aligned} -\partial_t \tau(t) - m(t,p)\tau(t) + \bar{I} h(t,p) = 0 \quad \tau(T) = 0 \\ \bar{\rho}_1 = -\int_0^T \tau(t) m_1(t) \xi(t)\,\mathrm{d}t \\ \bar{\rho}_2 = -\int_0^T \tau(t) m_2(t) \xi(t)\,\mathrm{d}t \end{aligned} \tag{B.48}$$

Thereby, the adjoint state variable is $\tau(t)$. The adjoint state equation that governs $\tau(t)$ can analogously to Equation (B.38) be derived using integration by parts, similarly the initial value $\tau(T) = 0$. The adjoints $\bar{\rho}_1$ and $\bar{\rho}_2$ are derived from $\frac{\partial g(\xi,p)}{\partial \rho_i} = -m_i(t)\xi(t)$.

B.2.6 Differentiation Method for the K-Ratio Model

Now, we extend the differentiation method from the simplified model in the previous section to the full k-ratio model from Chapter A. Of special focus in the energy stepping for the solution of the P_N-equation. We interpret the moments ψ as being already discretized on the grids $G_{\dots}$ defined in Section A.2.1 and do not specifically deal with the spatial integration in Equation (A.7). In Equation (A.63) the spatial integral is reduced to a sum, hence an adjoint version of the spatial integration is straightforward. Also, the mass concentrations ρ are interpreted as evaluated at the grid points $x \in G_{\dots}$.

The evolution of the discretized moments ψ in energy can be viewed as the iterative application of the operator

$$\psi_{i+1} = P_N^{i \to i+1}(\psi_i, \rho). \tag{B.49}$$

The operator depends on the discretized mass concentrations ρ and computes the moments ψ_{i+1} from ψ_i. While computing the moments ψ_i, simultaneously the ionization distribution $\phi = \{\phi_{(Z,j)}, \dots\}$ is approximated using the trapezoidal integration rule (Equation (A.60)). The iterative accumulation of the integral can be written as

$$\phi_{i+1} = \phi_i + \Delta\varepsilon_i \Phi_i, \tag{B.50}$$

where we define $(\Phi_{(Z,j)})_i = \sigma_{(Z,j)}(\epsilon_i)(\psi_0^0)_i$. For a convenient, iterative notation of the trapezoidal rule, we define additional energy steps $\varepsilon_i = \epsilon_{i-1} + \frac{\Delta\epsilon_{i-1}}{2}$ between the original steps ϵ_{i-1} and ϵ_i. Initial $\varepsilon_1 = \epsilon_1$ and cutoff energy $\varepsilon_{n_\epsilon+1} = \epsilon_{n_\epsilon}$ are exceptions. Analogously we define the step $\Delta\varepsilon_i = \varepsilon_{i+1} - \varepsilon_i$. The iterative definition in Equation (B.50) coincides with the trapezoidal rule in Equation (A.60). We interpret $(\Phi_{(Z,j)})_i$ as the mean of the integrand over the interval $[\varepsilon_i, \varepsilon_{i+1}]$

$$(\Phi_{(Z,j)})_i \approx \frac{1}{\Delta\varepsilon_i} \int_{\varepsilon_i}^{\varepsilon_{i+1}} \sigma_{(Z,j)}(\epsilon)\psi_0^0(\epsilon)\,\mathrm{d}\epsilon. \tag{B.51}$$

After the last iteration n_ϵ of computing the moments ψ_i and their integration into ϕ_i, ϕ_{N_e} is the ionization distribution, which is used to compute intensities I.

With special focus on the P_N energy stepping (Equation (B.49)) and the integration (Equation (B.50)), we visualize the computational graph of the k-ratio model in Figure B.3. Some dependencies in the computational graph are not discussed in this section: the relation between model parameters and mass concentrations $\rho(x; p)$ and the relation of the intensities I and y, an imaginary output scalar, to the ionization distribution ϕ_{n_ϵ}. But the generality in both relations is exactly the abstraction we try to achieve; we want to encapsulate the application of the adjoint state method only to the part of the forward model, where its application is necessary. The part of the forward model where adjoint mode algorithmic differentiation cannot be conveniently applied. Hence, the generality in the parts of the computation prior to mass concentrations ρ and after intensities I is intended.

Abusing the notation of AD, we will now derive the adjoint propagations inside the P_N-model as an interface between adjoint algorithmic differentiation and the adjoint state method. Recall the relation of derivative and adjoint (Equation (B.45)) from the previous section.

We discuss the adjoint of the operator $\psi_{i+1} = P_N^{i \to i+1}(\psi_i, \rho)$. In Figure B.3 the connection between the solution variable ψ_i and the adjoint state variable $\lambda_{n_\epsilon - i+1}$ is apparent. For every

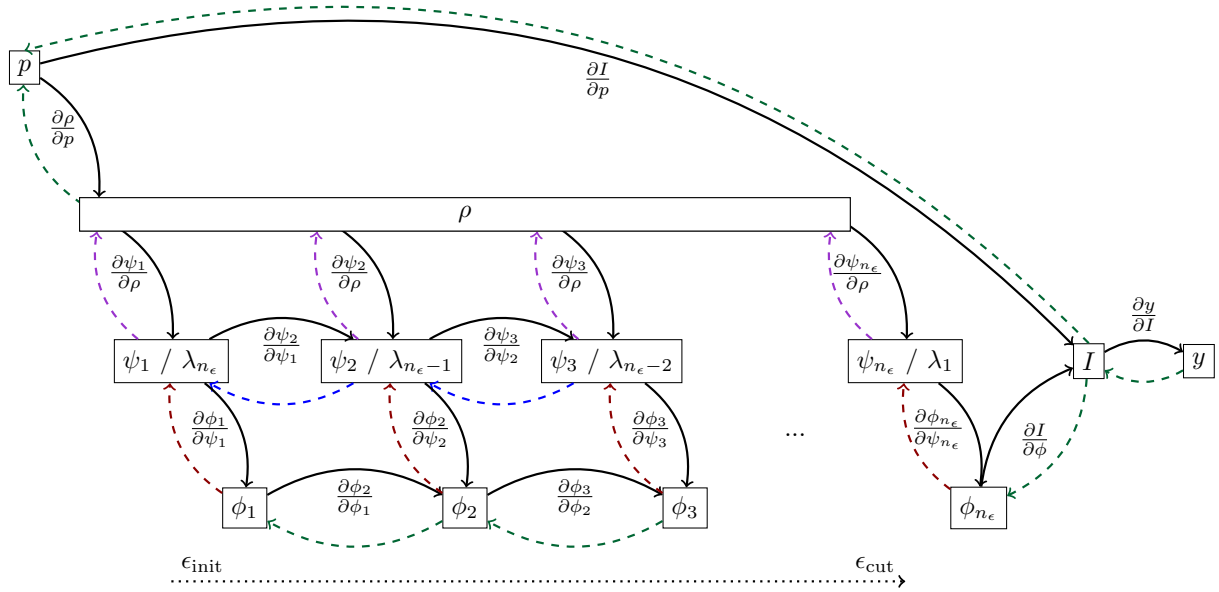


Figure B.3: *The computational graph of the calculation of the forward problem with special focus on the energy stepping while solving the P_N-equation and the iterative integration into the ionization distribution. From material parameters $\mathbf{p}$, a mass concentration field ρ is computed. Every energy step of the P_N equation $\psi_1, \dots \psi_{\mathbf{n_\epsilon}}$ depends on the mass concentrations ρ and the previous moments ψ_{i-1}. After each step the moments are iteratively integrated into the ionization distribution $\phi_1, \dots \phi_{\mathbf{n_\epsilon}}$. The dotted line $\epsilon_{init} \rightarrow \epsilon_{cut}$ symbolizes the evolution of the moments from high to low energies during the forward problem. The last iterate of the integration $\phi_{\mathbf{n_\epsilon}}$ is the complete ionization distribution and is used to compute intensities I and an imaginary output scalar y. The direct relation of intensity I to material parameters $\mathbf{p}$ due to mass attenuation and x-ray generation distribution is not detailed here. Dashed arrows symbolize the adjoint computation. Green dashed arrows are not detailed in this section and are left to an AD tool. Red, blue and purple dashed arrows are governed by the adjoint state equation and are detailed in this section. Note the reversed computation of the adjoint state variable λ_i from low (ϵ_{cut}) to high (ϵ_{init}) energies (blue dashed arrow).*

update of the solution variable $\psi_i \to \psi_{i+1}$ we identify the update of the adjoint state variable $\lambda_{n_\epsilon - i} \to \lambda_{n_\epsilon - i+1}$ as the adjoint version of $P_N^{i \to i+1}(\psi_i, \rho)$.

If we interpret the adjoint state variable $\lambda_{n_\epsilon - i+1}$ as the adjoint of the solution variable $\bar{\psi}_i$, we can identify the computation of the adjoint $\bar{\psi}_i$ (abusing the notation of AD)

$$\bar{\psi}_i = \left(\frac{\psi_{i+1}}{\psi_i}\right)^T \bar{\psi}_{i+1} \quad \leftrightsquigarrow \quad \lambda_{n_\epsilon - i+1} = P_N^{*, n_\epsilon - i \to n_\epsilon - i+1}(\lambda_{n_\epsilon - i}, \rho, \left(\frac{\partial f}{\partial \psi}\right)_i), \tag{B.52}$$

with the evolution of the adjoint state variable using $P_N^{*, i \to i+1}$. Analogously to Equation (B.49) also λ is interpreted as already discretized on the grids $G_{\dots}$.

The source in the adjoint state equation $\left(\frac{\partial f}{\partial \psi}\right)_i$ relates to the update $\psi_i \to \psi_{i+1}$, resp. $\lambda_{n_\epsilon - i} \to \lambda_{n_\epsilon - i+1}$. Thereby the function f contains all computations up to the output scalar y. Also the computations we are trying to keep general. However, using Equation (B.45) we can write

$$\left(\frac{\partial f}{\partial \psi}\right)_i = \left(\frac{\partial \Phi_i}{\partial \psi_l^k}\right)^T \bar{\Phi}_i. \tag{B.53}$$

We base the calculation of the source of the adjoint state equation on the adjoint $\bar{\Phi}_i$ and only deal with $\left(\frac{\partial \Phi_i}{\partial \psi_l^k}\right)^T$. The computation of the adjoint $\bar{\Phi}_i$ can be dealt with by an algorithmic differentiation tool. For completeness, we define the adjoint $\bar{\Phi}_i$ as

$$\bar{\Phi}_i = \Delta\varepsilon_i \bar{\phi}_i = \Delta\varepsilon_i \bar{\phi}, \tag{B.54}$$

where the adjoint of the ionization cross-section $\bar{\phi}_i = \bar{\phi}$ because of the constant propagation of adjoints through the sum in Equation (B.50) ($\frac{\partial \phi_{i+1}}{\partial \phi_i} = 1$). Basing the computation of the source on the adjoint $\bar{\Phi}_i$ (or $\bar{\phi}$) achieves generality in the computations after the ionization distribution.

To derive the explicit form of Equation (B.53), we analyze Equation (B.51). For each computation of $\Phi_{(Z,j),i}$ we weight ψ_0^0 by $\sigma_{(Z,j)}$, hence the adjoint version is given by

$$\left(\frac{\partial f}{\partial \psi_l^k}\right)_i = \left(\frac{\partial \Phi_i}{\partial \psi_l^k}\right)^T \bar{\Phi}_i = \sum_{(Z,j)} \frac{1}{\Delta\varepsilon_i} \sigma_{(Z,j)} \delta_{l,0} \delta_{k,0} \bar{\Phi}_{(Z,j),i} \tag{B.55}$$

The source only applies to λ_0^0 (c.f. the deltas $\delta_{l,0}$ and $\delta_{k,0}$), because only ψ_0^0 is considered to compute Φ_i.

Generality in the parametrization of $\rho(x; p)$ is achieved by a similar approach. The adjoint of the mass concentrations are $\bar{\rho} = \frac{\partial y}{\partial \rho}$ (Equation (B.45)). Hence, the adjoints $\bar{\rho}$ are governed by the last step of the adjoint state method (Equation (B.37))

$$\bar{\rho} = \frac{\partial y}{\partial \rho} = \frac{\partial}{\partial \rho} \langle \lambda, g(\rho, \hat{\psi}) \rangle \tag{B.56}$$

The scalar product in Equation (B.56) is the inner product in the function space L^2. To derive Equation (B.56) in explicit form, recall the P_N-constraint $g(\rho, \psi)$ from Equation (A.21) and the definitions of stopping power $S = \sum_{i=1}^{n_e} \rho_i S_i$ (Equation (A.23)) and transport coefficient $C = \sum_{i=1}^{n_e} \rho_i C_i$ (Equation (A.26)). The advection matrices in $g(\rho, \psi)$ do not depend on

the mass concentrations ρ, therefore

$$\begin{aligned}\bar{\rho} = \frac{\partial}{\partial \rho}\langle \lambda, g(\rho, \hat{\psi})\rangle &= \frac{\partial}{\partial \rho}\int_{\epsilon_{\text{cut}}}^{\epsilon_{\text{init}}} \lambda^T(-\partial_\epsilon(S\hat{\psi}) + C\hat{\psi})\,\mathrm{d}\epsilon \\ &\approx \sum_{i=1}^{n_\epsilon} \lambda_{n_\epsilon - i+1}^T(-\partial_\epsilon(\frac{\partial S}{\partial \rho}\hat{\psi}_i) + \frac{\partial C}{\partial \rho}\hat{\psi}_i)\Delta\varepsilon_i.\end{aligned} \tag{B.57}$$

The integral in energy is again approximated by the trapezoidal rule using the energy steps $\Delta\varepsilon_i$. The derivatives of the material properties $\frac{\partial S}{\partial \rho_k} = S_k$ and $\frac{\partial C}{\partial \rho_k} = C_k$ are the coefficients of pure elements k, because of the linearity of the sums.

We identify each of the summands in Equation (B.57) with the adjoint accumulation of $\lambda_{n_\epsilon - i+1}$ into $\bar{\rho}$, therefore (abusing the notation of AD)

$$\bar{\rho}_k \mathrel{+}= \lambda_{n_\epsilon - i+1}^T(-\partial_\epsilon(S_k\hat{\psi}_i) + C_k\hat{\psi}_i)\Delta\varepsilon_i \quad \leftrightsquigarrow \quad \bar{\rho}_k \mathrel{+}= \left(\frac{\partial \psi_i}{\partial \rho_k}\right)^T \bar{\psi}_i. \tag{B.58}$$

Equation (B.58) is the analogue to the adjoint accumulation of $\bar{\psi}_i \leftrightsquigarrow \lambda_{n_\epsilon + i+1}$ into $\bar{\rho}_k$ in algorithmic differentiation.

B.2.6.1 Implementation Example

While being conceptually different, the adjoint state method and algorithmic differentiation share many similarities. Based on the analogies presented in the previous section, we address the similarity in their implementation. Therefore, we provide a basic implementation example of the described differentiation method following the custom adjoint implementations in Section B.2.3. The code is given in Figure B.4. While hiding many implementation details, it provides a basic overview of the method when applied in an implementation. The primal function evaluation `calc_`ϕ (initialization, P_N-stepping and integration) is augmented with storing the computed values of ψ to a tape ψ`_tape`, a step that is only necessary for the adjoint computation, as the primal integration to ϕ can be implemented alongside the iterations to solve ψ. Memorizing intermediate variable values on a tape is a common pattern also in algorithmic differentiation.

In the adjoint function `calc_`ϕ`_adjoint`, the adjoint state variable λ is initialized, the source of the adjoint state equation `dfdpsi` is computed using the given adjoint ϕ`_adjoint` and used to perform the stepping in the adjoint P_N-equation. Additionally, ψ is retrieved from the tape ψ`_tape` and is used together with λ to accumulate the adjoint ρ`_adjoint`.

```
@adjoint function calc_ϕ(ρ)
    ψ = ψ0()
    ϕ = 0

    ψ_tape = []
    push!(ψ_tape, ψ)
    ϕ = ϕ + integrate_ϕ(ψ)
    ϵ = ϵ_init
    while ϵ > ϵ_cut
        ψ, ϵ = step_pn(ψ, ρ, ϵ)
        push!(ψ_tape, ψ)
        ϕ = ϕ + integrate_ϕ(ψ)
    end

    function calc_ϕ_adjoint(ϕ_adjoint)
        λ = λ0()
        ρ_adjoint = 0

        ϵ = ϵ_cut
        while ϵ < ϵ_init
            ψ = pop!(ψ_tape)
            ρ_adjoint += integrate_ρ_adjoint(λ, ψ)
            dfdpsi = compute_adjoint_source(ϕ_adjoint)
            λ, ϵ = step_pn_adjoint(λ, ρ, dfdpsi, ϵ)
        end
        ψ = pop!(ψ_tape)
        ρ_adjoint += integrate_ρ_adjoint(λ, ψ)

        return ρ_adjoint
    end
    return ϕ, calc_ϕ_adjoint
end
```

Figure B.4: *Implementation of the differentiation method described in Section B.2.6. While hiding many implementation details, it provides an overview over an implementation of the method.*

B.3 Reconstruction Experiments

B.3.1 Showcase: Inversion of Mass Concentration Models

We compare the mass concentration models presented in Section A.3.1 on the basis of their properties when applied in a reconstruction.

Consider a compound of lead Pb, silicon Si and copper Cu. We define the objective to compute mass fractions ω_i, volume fraction φ_i and the parameters in the linear density model γ_i from given mass concentrations ρ_i^{true}, which are computed using $\{\omega, \varphi, \gamma\}_i = \frac{1}{3}$ for $i = \{Pb, Si, Cu\}$. In Figure B.5 the value of the squared error $||\rho(\{\omega_i, \varphi_i, \gamma_i\}) - \rho^{\text{true}}||^2$ is illustrated for varying parameters of Pb and Si (Cu can be deduced).

The computation of mass fractions from mass concentrations resembles the last subproblem of the actual problem of reconstruction in EPMA. Although it is never explicitly formulated in the actual reconstruction problem, it appears overshadowed by the subsequent computation of the forward model. We mention the computation of mass fractions from mass concentrations to shows the importance of formulating a proper material parametrization that behaves well-posed in a reconstruction. If the material parametrization already shows ill-posed behavior, we cannot expect the actual reconstruction to provide reliable results.

All the mass concentration models compared in Figure B.5 show a unique minimum, but a differently shaped squared error function. We would observe different convergence behavior of e.g. the gradient descent method. The method performs best, if the optimum is equally curved in all directions, which is not the case in either of Figure B.5.

A measure for equal curvature (which additionally determines the speed of convergence of the gradient-descent method) is the condition number κ of the Hessian $H_{\{\omega,\varphi,\gamma\}}||\rho(\{\omega, \varphi, \gamma\}_i = \frac{1}{3}) - \rho^{\text{true}}||^2$. The condition number κ is given by the ration of maximal to minimal eigenvalue of the Hessian (c.f. Section B.1.2). For the squared errors of mass concentrations we compute

$$\kappa(H_\omega) \approx 2.34 \quad \kappa(H_\varphi) \approx 5.39 \quad \kappa(H_\gamma) \approx 2.41, \tag{B.59}$$

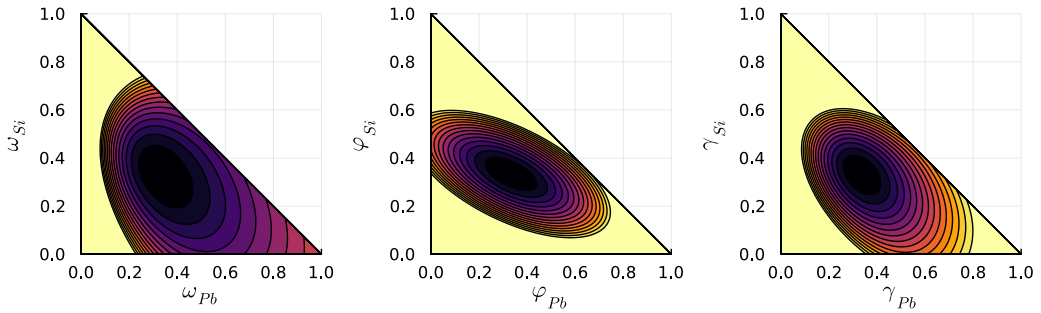
and quantify the visual observation in Figure A.4. Even if the condition number of the volume fraction model is higher compared to the mass fraction and the linear density model, a condition number $\kappa(H_\varphi)$ of 5.39 is moderate. Hence, all models are suited for application in an actual reconstruction problem in EPMA, if the assumptions, which are made in Section A.3.3 apply for the compound.

B.3.2 Comparison: Sensitivities of a 1D Material with Finite Differences

We compare sensitivities $\frac{\partial I_{(Z,j)}}{\partial p_k}$ of intensities of multiple x-rays (Z, j) with respect to model parameters computed using the differentiation method described in Section B.2.6 with sensitivities computed by second order central finite differences. The sensitivities are given by the partial derivative of the intensity, which is computed using the model presented in Section A.1, with respect to the parameters p_k

$$\frac{\partial I_{(Z,j)}}{\partial p_k} = \frac{\partial}{\partial p_k} \mathcal{A}_{(Z,j)}(\rho_i(x; p_k)) \circ \mathcal{X}_{(Z,j)}(\rho_i(x; p_k)) \circ \phi_{(Z,j)} \circ \psi(\rho_i(x; p_k)). \tag{B.60}$$

A comparison to finite differences addresses several questions at the same time:



(A) *Mass fractions* (B) *Volume fraction* (C) *Linear density*

Figure B.5: *The squared error between mass concentrations* $||\rho(\{\omega,\varphi,\gamma\}_i) - \rho^{true}(\{\omega,\varphi,\gamma\}_i = \frac{1}{3})||^2$ *in a* $\{Pb, Si, Cu\}$*-compound computed using the different mass concentration models. Mass fractions* ω_i*, volume fractions* φ_i *and* γ_i *are constrained by Equation* (A.69)*, hence the triangular shape. For better visualization of the minimum, the color values are clipped to* 0 *: black,* > 10 *: yellow.*

domain $\mathfrak{S}$	$(-500\text{nm}, 0\text{nm})$ $n_x = 100$
energies $[\epsilon_{\text{cut}}, \epsilon_{\min}]$	$(0.5\text{keV}, 16\text{keV})$
approximation order P_N	$N = 9$
beam $\epsilon, \sigma_\epsilon$	15keV, 0.1keV
elements	Cu and Ni, $\{\omega,\varphi,p\}_i = 0.5$

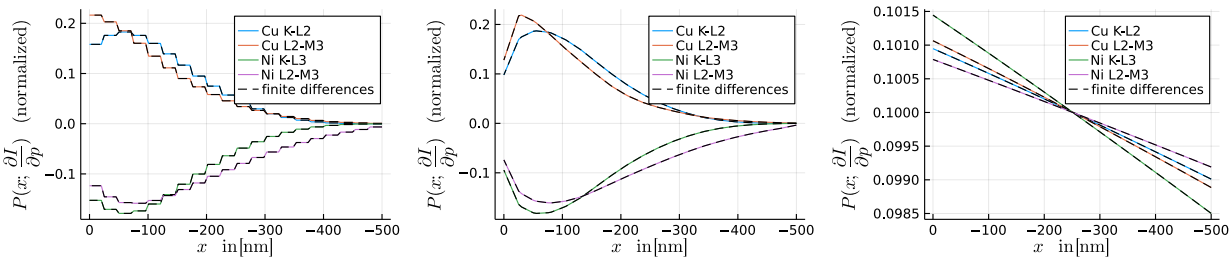
Table B.2: *Model settings used for the comparison of 1D sensitivities*

- If the computed derivatives between our method and finite differences match, the comparison validates our implementation.
- We can demonstrate the claimed superiority of our method compared to finite differences in terms of computation time.
- It provides insight into the nature of to model. The sensitivities $\frac{\partial I_{(Z,j)}}{\partial p_k}$ are the first order linear dependency of intensities with respect to model parameters, hence a large sensitivity relates to a strong dependency, a small sensitivity to a weak dependency. This is particularly interesting for the inverse problem, because it also unveils possible difficulties in the reconstruction.

For validation of our implementation, we compare the sensitivities for all parametrizations introduced in Section A.3.3. The physical setup and the settings of the P_N approximation for this example can be found in Table B.2 while the configurations of the parametrizations are given in Table B.3. Parameter values for the parametrizations are randomly chosen.

Piecewise-Constant	$(n_x = 21, n_y = 2, n_z = 2)$, $n_p = 20$
Linear	$(n_x = 20, n_y = 1, n_z = 1)$, $n_p = 20$
Neural Network	3 layers, $(1 \times 1, \text{norm.}) \rightarrow (1 \times 10, \tanh) \rightarrow (10 \times 1, \text{sigmoid})$, $n_p = 31$

Table B.3: *Parametrization used for the comparison of 1D sensitivities*



(A) *Piecewise-Constant* (B) *Linear* (C) *Neural Network*

Figure B.6: *A visualization of the sensitivities computed using the method presented in Section B.2.6 and using finite differences. We abuse the parametrization function of each parametrization and plot* $P(x; \frac{\partial I_{(Z,i)}}{\partial p_k})$. *The curves computed using our method (solid lines) coincide with the curves computed using finite differences (black dashed lines). While for the piecewise-constant and the linear parametrization an interpretation of this plot is possible, an interpretation for the neural network is unclear. A quantitative comparison of sensitivity values is given in Table B.4.*

For a visualization of sensitivities, we abuse the parametrization functions $P(x; p)$ which are defined for each parametrization. The sensitivities of one x-ray intensity $I_{(Z,j)}$ have the same dimensionality as the parameters describing the material, hence we visualize the output of the material parametrization defined in Section A.3.3 using the sensitivities as its parameters $P(x; \frac{\partial I_{(Z,j)}}{\partial p_k})$. In Figure B.6 the parametrization functions $P(x; \frac{\partial I_{(Z,j)}}{\partial p_k})$ are visualized.

For the piecewise-constant and the linear parametrization an interpretation of this plot is possible. Because the function value $P(x; p)$ for piecewise-constant and linear is the value of the parameters p (or a linear interpolation of it), we have an intuition how to interpret the plot. Increasing a parameter, increases the mass concentration of Cu and simultaneously decreases the mass concentration of Ni, hence the positive values for Cu x-rays and the negative value for Ni x-rays. In depth the sensitivity decreases, because the ionization distribution for all x-rays decreases and the absorption increases.

However, for the neural network $P(x; p)$ is the forward evaluation of the network based on the sensitivities, and we lack a clear interpretation.

For each parametrization we compare $P(x; \frac{\partial I_{Z,j}}{\partial p})$ for sensitivities computed by our method (colored, solid lines) with the sensitivities computed by finite differences (black, dashed lines). The curves coincide. Additionally, since the visualizations are overshadowed by the parametrization function $P(x; p)$ where errors in the sensitivities can get cancelled, we compared the relative errors of sensitivities

$$\max_k \left| \frac{1}{\frac{\partial I_{(Z,j)}}{\partial p_k}} \left(\frac{\partial I_{(Z,j)}}{\partial p_k} - \frac{I_{(Z,j)}(p_k + h) - I_{(Z,j)}(p_k - h)}{2h} \right) \right|. \tag{B.61}$$

In Table B.4 the relative errors are shown. All relative errors are < 0.003. Derivatives computed by algorithmic differentiation are exact (up to machine precision) [Naumann, 2011]. Our method as well as finite differences only provide approximations of the derivative. For our method, the underlying approximation of the adjoint state variable introduces errors, for finite differences the approximation error originates from the step size h. The presented

$P(x;p)$ \ (Z,j)	$(Cu, K-L2)$	$(Cu, K-L2)$	$(Cu, K-L2)$	$(Cu, K-L2)$
Piecewise-Constant	0.0015	0.0005	0.0013	0.0003
Linear	0.0030	0.0007	0.0008	0.0011
Neural Network	0.0001	5.9367×10^{-5}	6.1947×10^{-5}	4.8781×10^{-5}

Table B.4: *Relative Errors in the sensitivities/derivatives of intensities for multiple x-rays (Z, j) and parametrization methods $P(x; p)$. Sensitivities/derivatives are computed using the method presented in Section B.2.6 and using a finite difference approximation of order 2.*

$P(x;p)$ \ Method	AD	FiniteDiff
Piecewise-Constant	5.52s	31.02s
Linear	5.37s	31.14s
Neural Network	5.85s	45.84s

Table B.5: *Runtimes of the computation of sensitivities/derivatives using the method presented in Section B.2.6 (AD) and a finite difference approximation of order 2 (FiniteDiff).*

relative error of < 0.003, and various other comparisons of derivatives of individual model parts to finite differences which are performed during the implementation, convince us that the method correctly estimates derivatives.

In Table B.5 we compare the runtime of the computations. Here, the superiority of our method compared to finite differences becomes apparent. In Section B.2.1 we claimed that the differentiation using our method scales in the number output variables, while the application of finite differences scales in the number of input variables. The number of output variables is the number of different x-ray intensities $I_{(Z,j)}$, here four. The number of parameters n_p for each parametrization is given in Table B.3.

Approximating the sensitivities of the neural network parametrization using the presented method only takes marginally longer than approximating sensitivities of the piecewise-constant and the linear parametrization, although the neural network has $\approx 1.5\times$ the number of parameters. Using finite differences the scaling in the number of input parameters is clearly visible.

B.3.3 Showcase: 1D Reconstruction of a Sharp and a Diffusive Interface

We present the reconstruction of a coated material consisting of iron Fe and nickel Ni with a sharp (discontinuous) and a diffusive (continuous) interface between the layers. By means of this example we compare the parametrizations described in Section A.3.3. For a fair comparison, the number of parameters in each parametrization is chosen to be 10. For the piecewise-constant parametrization, the interval $[0\text{nm}, -300\text{nm}]$ is divided in depth into 10 equal parts. Note that the location of the sharp interface aligns with one of the interfaces of the piecewise-constant parametrization. For the linear interpolation, we discretize the interval in 9 equal parts. The neural network consists of a normalization layer, a dense 1×3

tanh layer and a 3×1 sigmoid layer. We strongly enforce the constraint for the volume fractions $\varphi_{Ni} = 1 - \varphi_{Fe}$, hence at every location a scalar $P(x; p)$ completely describes the material. To resolve the interface position and structure, we compare k-ratios from multiple different beam energies between 9keV and 15keV. Additional P_N model settings and the considered x-rays are tabulated in Table B.6. We use the squared error of k-ratios as the objective function and choose the BFGS method implemented in `Optim.jl` [Mogensen and Riseth, 2018] as optimization method.

In Figure B.7 the reconstructed density of the material with a sharp interface is visualized using the different parametrizations. From initial configurations (piecewise-constant, linear: 0.5 Fe and 0.5 Ni; neural network: random) the parametrizations iterate towards the reference density (black line) during the optimization. We visualize the initial, first, fifth iteration and the 100th iteration. The inability of the linear parametrization to represent discontinuities becomes apparent. However, the piecewise-constant parameterization only approximates perfectly because the interface matches the interface of the reference. On the other hand, the neural network parametrization is flexible enough to identify the location of the interface and to approximate the discontinuity.

In Figure B.8 the reconstructed density of the material with a diffusive interface is visualized. Except for the reference material, exactly the same settings as for the reconstruction of the sharp interface are used. Again, the parametrizations iterate towards the reference density (black line). For the diffusive interface, none of the parametrizations can reconstruct the interface perfectly, but the linear and the neural network parametrizations perform better than the piecewise-constant parametrization.

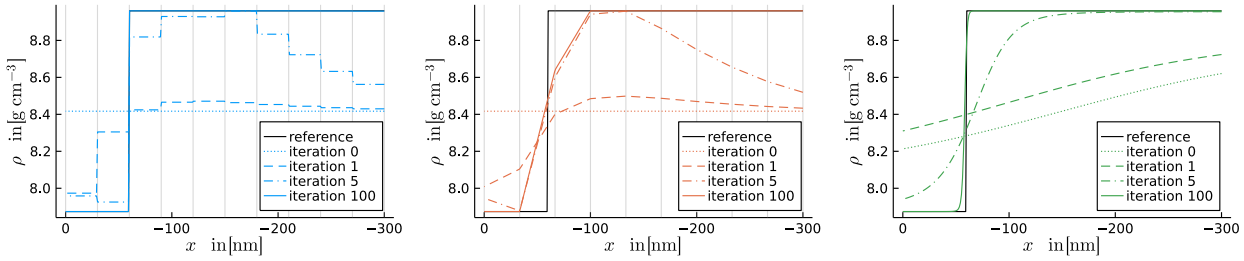
In Figure B.9 we visualize the normalized error, the value of the objective function $||k(p) - k^{\text{exp}}||^2$, the error in the mass concentrations $||\rho(x) - \rho^{\text{true}}||^2$ and the error of additional k-ratio measurements $||k_+(p) - k_+^{\text{exp}}||^2$ (with beam energies 11keV and 14keV) during the optimization iterations. The kinks in the error of the objective function are an artifact of the `Optim.jl` implementation. Their default implementation bases on the combination of a line search method [Hager and Zhang, 2005] and the BFGS algorithm presented in Nocedal and Wright [2006].

Differences in the performance of the different parametrizations are clearly visible. The material error for the piecewise-constant parametrization for the sharp interface is by far the smallest, due to the possible perfect reconstruction. Comparing the material error for piecewise-constant and linear parametrization for the sharp and the diffusive interface, they vary. Only the neural network parametrization performs similarly well for both examples.

A correlation between the material error and the error of additional k-ratios would be beneficial, but is hard to obtain. Ideally both would behave similarly, because then we could propose the error in additional k-ratios as a suitable reconstruction quality measure (Section B.1.1). However, the evaluation of reconstruction quality measures is beyond the scope of this work and is deferred to further research.

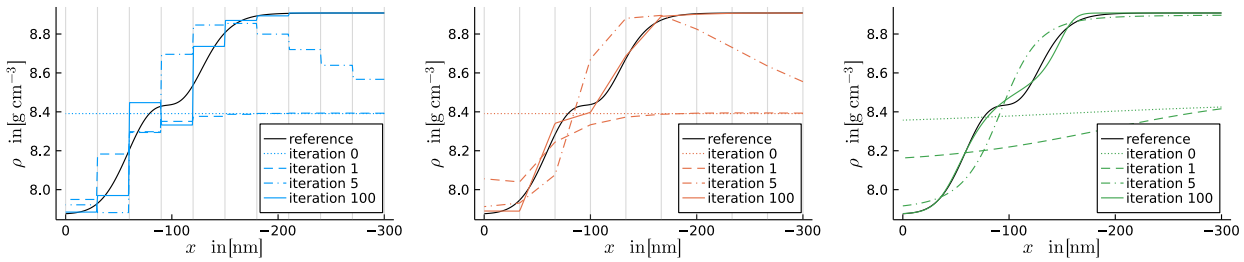
domain $\mathfrak{S}$	$(-500\text{nm}, 0\text{nm})$ $n_x = 100$
energies $[\epsilon_{\text{cut}}, \epsilon_{\min}]$	$(7.0\text{keV}, 15.5\text{keV})$
approximation order P_N	$N = 9$
beam $\epsilon, \sigma_\epsilon$	$\{9, 10.5, 12, 13.5, 15\}\text{keV}$, 0.1keV
elements/x-rays	$(Ni, K-L_2)$, $(Ni, K-L_3)$, $(Fe, K-L_2)$ and $(Fe, K-L_3)$

Table B.6: *Model settings used for the reconstruction of the 1D interfaces*



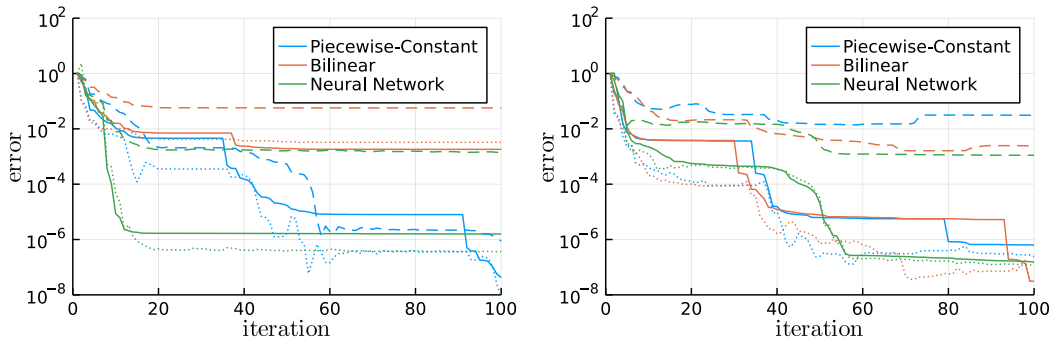
(A) *Piecewise-Constant* (B) *Linear* (C) *Neural Network*

Figure B.7: *The reconstructed density of a material with a sharp interface between an iron Fe layer covering an Ni substrate. We use multiple material parametrizations presented in Section A.3.3 for the reconstruction. For the piecewise-constant and the linear parametrization, the geometry (interfaces) are visualized by gray vertical lines. The black line shows the reference density. All parametrizations converge. The piecewise-constant parametrization has a clear advantage, because the interface aligns with an interface of the parametrization. The linear parametrization cannot approximate the discontinuous interface, hence it converges to presented smoothed representation. The neural network is flexible enough to identify the location of the interface and also approximates the discontinuity using a strong gradient.*



(A) *Piecewise-Constant* (B) *Linear* (C) *Neural Network*

Figure B.8: *The reconstructed density of a material with a diffusive interface between an iron Fe layer covering an Ni substrate. We use multiple material parametrizations presented in Section A.3.3 for the reconstruction. For the piecewise-constant and the linear parametrization, the geometry (interfaces) are visualized by gray vertical lines. The black line shows the reference density. All parametrizations converge. None of the parametrizations can represent the interface perfectly. The bilinear and the neural network perform better for this example, because of their flexibility to approximate continuous functions.*



(A) *Sharp Interface* (B) *Diffusive Interface*

Figure B.9: *The value of the objective function and additional reconstruction quality measures during the iterations of the optimization for all parametrizations. The objective function* $||k(p) - k^{exp}||^2$ *is visualized as a solid line. Also, the material error* $||\rho(x) - \rho^{true}||^2$ *(dashed line) and the error of k-ratios* $||k_+(p) - k_+^{exp}||^2$ *(dotted line) from additional beam energies (*11keV *and* 14keV*) is plotted.*

domain $\mathfrak{S}$	$(-300\text{nm}, 0\text{nm}) \times (-150\text{nm}, 150\text{nm})$ $(n_x, n_y) = (80, 80)$
energies $[\epsilon_{\text{cut}}, \epsilon_{\min}]$	$(7.0\text{keV}, 12.5\text{keV})$
approximation order P_N	$N = 9$
beam $\mu_\epsilon, \sigma_\epsilon$	12keV, 0.2keV
beam μ_x, σ_x	$(0, \{-50, -25, 0, 25, 50\})$nm, 20nm
beam μ_Ω, κ	$[-1, 0]$, 10
elements/x-rays	$(Cu, K-L_2)$, $(Cu, K-L_3)$, $(Fe, K-L_2)$ and $(Fe, K-L_3)$
standard materials	$(Cu, K-L_*) : (1.0Cu, 0.0Fe)$, $(Fe, K-L_*) : (0.0Cu, 1.0Fe)$

Table B.7: *Model settings used for the 2D reconstruction of the ellipsoidal Cu inclusion.*

B.3.4 Showcase: 2D Reconstruction of an Ellipsoidal Material Structure

We describe the use case of a reconstruction of an ellipsoidal copper Cu inclusion in an iron Fe substrate. With this example we also show the extensibility of our method to an additional material parametrization.

A researcher is investigating a material consisting of Cu and Fe, where he conducts experiments using a 12keV electron beam with multiple beam positions. From a line-scan he observes the (non-noisy) k-ratio profile plotted in Figure B.10. Only the k-ratios retrieved from five beam positions $\mu_y = \{-50, -25, 0, 25, 50\}$nm which are marked by black crosses are used for the reconstruction. The profile shows an increase in the $(Cu, K-L_2)$ k-ratio for a beam position close to $\approx$ 25nm. From previous analysis of the material the researcher knows, that the material tends to have elliptical Cu inclusions of different size, shape and interface structure. Hence, for the present investigation, he also assumes an elliptical material structure.

He specifies a material parametrization which encodes his knowledge of an elliptical structure (with unknown position (μ_1, μ_2), unknown rotation r and unknown scaling factors for the principal axes (a, b)). For the unknown interface structure, the neural network parametrization can be utilized in degenerated form. He specifies a parametrization (Section A.3.3.3) with the following layers: $(2 \times 2, \text{norm.}) \rightarrow (2 \times 1, \text{ellipse}) \rightarrow (1 \times 1, \text{sigmoid})$. The extension of the current implementation to a layer, which describes ellipsoidal structures is shown in Section C.2. To complete the material parametrization, he assumes that the volume fraction model (c.f. Section A.3.1) sufficiently describes the material. Hence, the scalar volume fraction $\varphi_{Cu}(x)$ is parametrized and the volume fraction of Fe is deduced from $\varphi_{Fe}(x) = 1 - \varphi_{Cu}(x)$. As an initial guess, he assumes the inclusion is circular and is located around the maximum of the $(Cu, K-L_2)$ k-ratio profile.

The k-ratio shown in the line profile in Figure B.10 are artificial (simulated) measurements. Hence, the reference material ρ^{true} which is used to compute the artificial measurements k^{exp} is known. We visualize the total density ρ_{tot} of the reference material in Figure B.11 and additionally show the ionization distributions of $(Cu, K{-}L_2)$ and $(Fe, K{-}L2)$ that define the size of the interaction volume. The ellipsoidal Cu inclusion is clearly smaller than the interaction volumes. Additional settings of the k-ratio model are shown in Table B.7.

For the reconstruction we use the L-BFGS algorithm implemented in `Optim.jl` [Mogensen and Riseth, 2018] using the default settings. The total density of the material ρ_{tot}

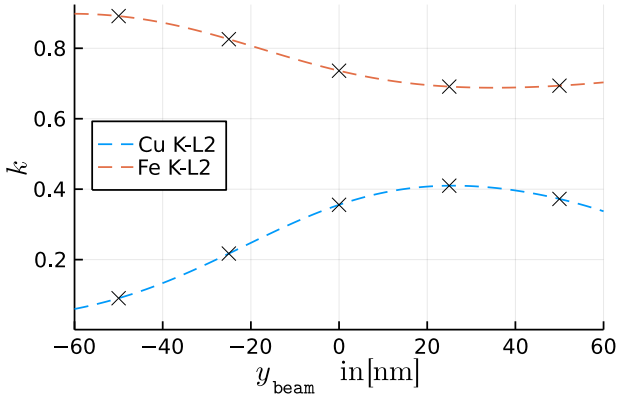
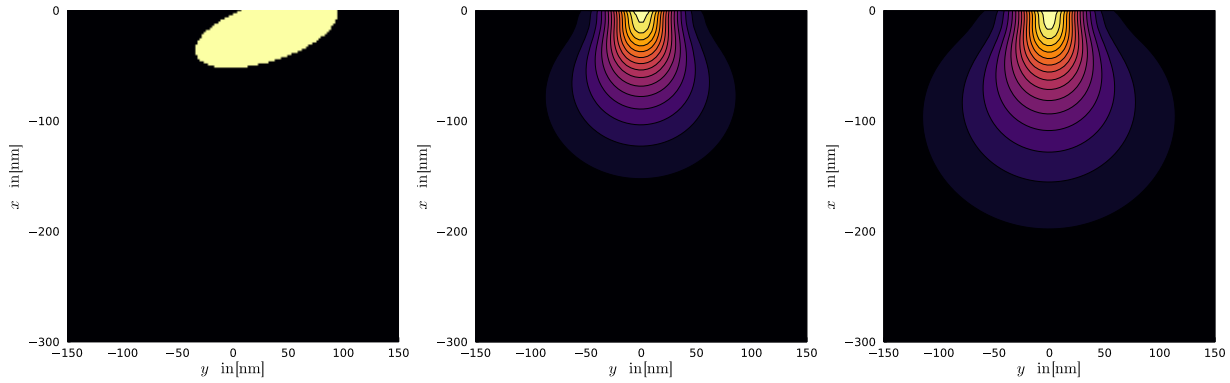


Figure B.10: *The k-ratio profile of the elliptical Cu inclusion in the Fe substrate computed using a* 12keV *electron beam. Dashed lines show the k-ratio profiles of* $(Cu, K-L_2)$ *(blue) and* $(Fe, K-L2)$ *(orange) with a high beam resolution. Black marker show the k-ratios which are used for the reconstruction. While not being directly visible, the shape and height of the k-ratio curves encode information about the shape and location of the inclusion and the structure of the interface between inclusion and substrate.*

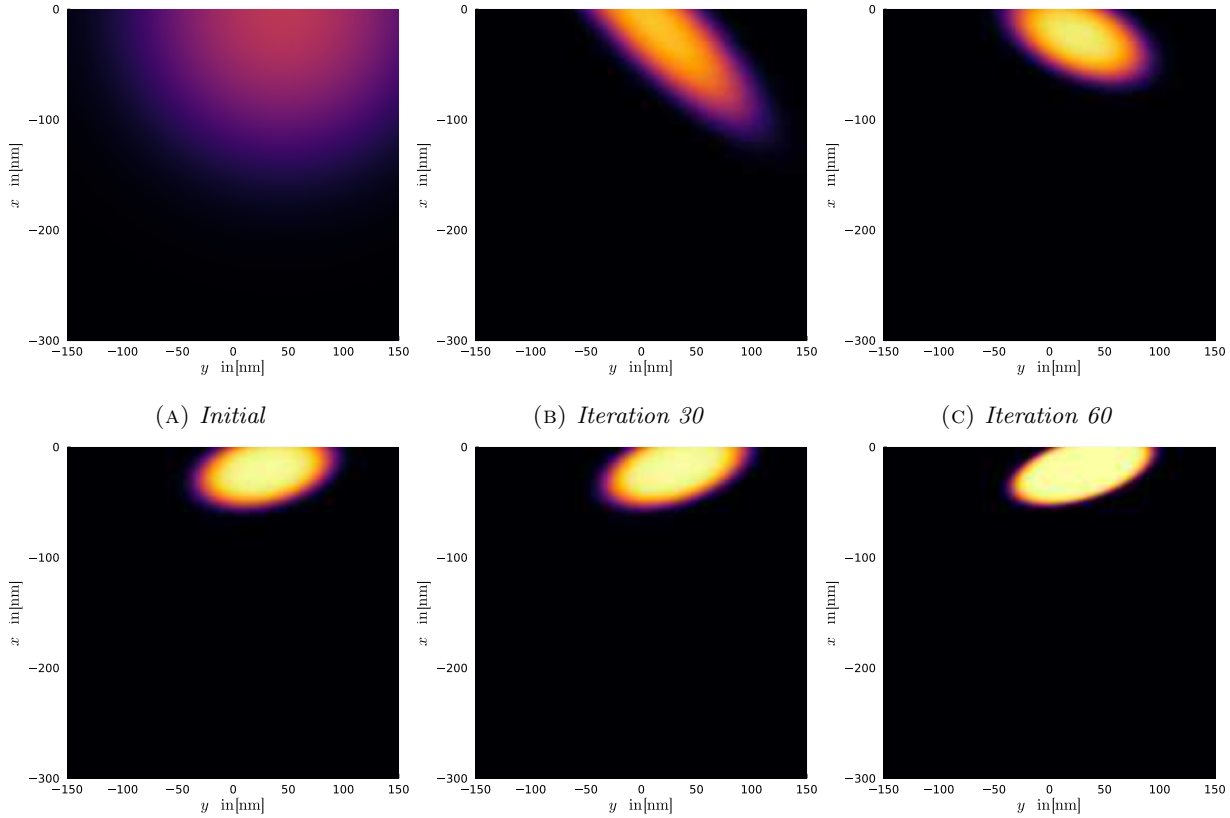


(A) *The density* ρ_{tot}^{true} *of the reference material.*

(B) *The ionization distribution* $\phi_{(Cu,K-L_2)}$.

(C) *The ionization distribution* $\phi_{(Fe,K-L_2)}$.

Figure B.11: *The total density and two ionization distribution fields of the reference material. The ionization distribution curves determine the size of the interaction volume. The size of the ellipsoidal Cu inclusion is smaller than the interaction volume.*



(a) *Initial* (b) *Iteration 30* (c) *Iteration 60*

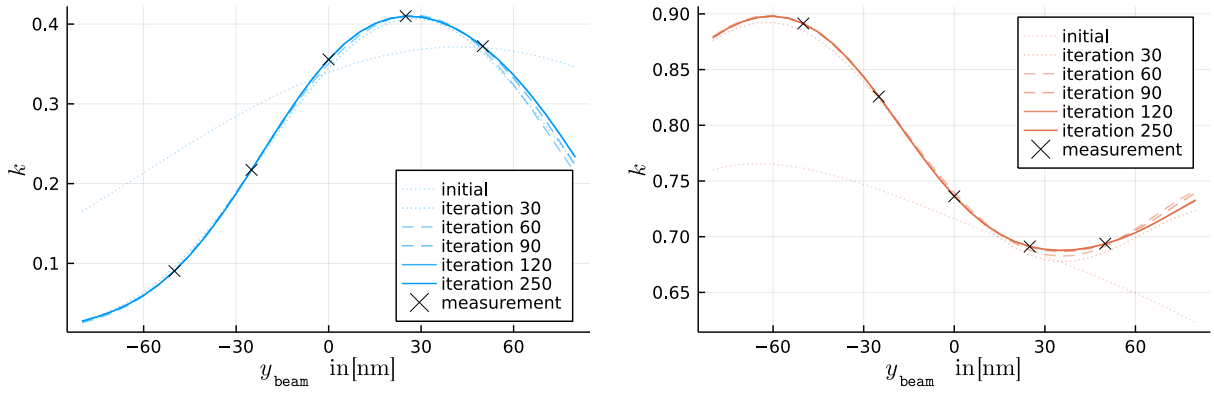
(d) *Iteration 90* (e) *Iteration 120* (f) *Iteration 250*

Figure B.12: *The reconstructed density ρ_{tot} of the Fe material with a Cu inclusion during the iterations of the L-BFGS optimization.*

during the iteration steps is visualized in Figure B.12. From the initial guess the reconstruction converges to the reference material. Alongside the total density of the reconstructed material we visualize the k-ratio profiles of $(Cu, K-L_2)$ and $(Fe, K-L_2)$ in Figure B.13. The shape of the curves quickly coincide with the measured k-ratios (black crosses). At iteration 30 the measured k-ratios are already well approximated, the total density, however, is not. For sufficient reconstruction of the ellipsoidal structure, the error in the k-ratios (the objective function) must be further reduced. In Figure B.14 we show the (normalized) value of the objective function $||k(p) - k^{\text{exp}}||^2$ and the (normalized) material error $||\rho(p) - \rho^{\text{true}}||^2$ during the iterations of the optimization. For the first 50 iterations, the value of the objective function decreases, whereas the material error remains close to the initial error. Only after the first 50 iterations, the material error also decreases. In Figure B.12 this effect is also observed, where only after iteration 60 the shape of the reference ellipse starts to form.

B.3.5 Comparison: Representation Capabilities of the Material Parametrizations

We compare the capability of each material parametrization presented in Section A.3.3 to represent the ellipsoidal material structure from Section B.3.4. Therefore, the material



(A) $(Cu, K - L_2)$ (B) $(Fe, K - L_2)$

Figure B.13: *The k-ratio profile of* $(Cu, K - L_2)$ *and* $(Fe, K - L2)$ *during the iterations of the reconstruction. Black crosses mark the measured k-ratios which are considered in the objective function* $||k(p) - k^{exp}||^2$ *(c.f. Figure B.10). The visualized iterations are the same as in Figure B.12. Note that the k-ratios quickly converge to the measured values, although the shape of the ellipse does not yet coincide with the shape of the ellipse in the true material (Figure B.11).*

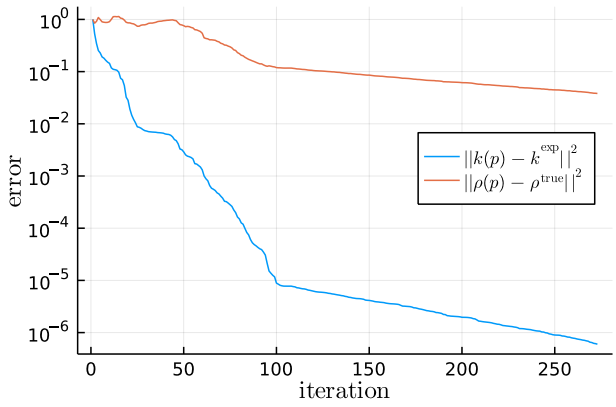
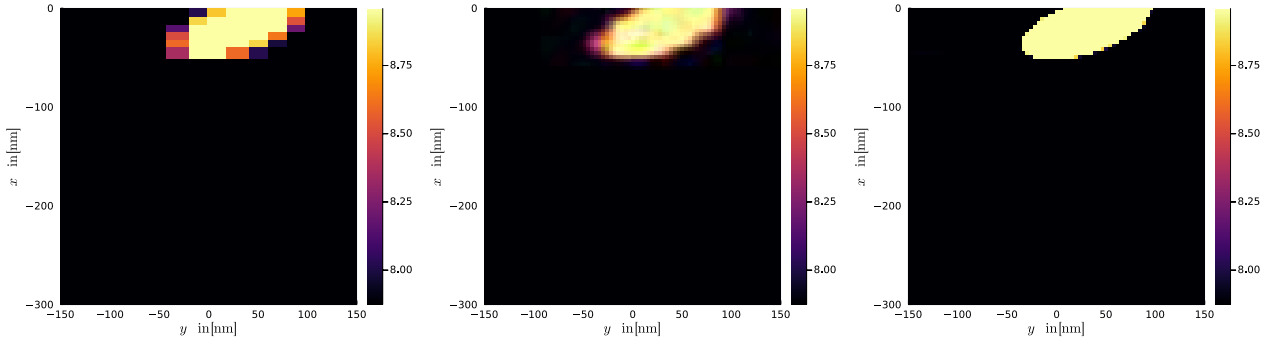


Figure B.14: *Normalized errors of the objective function* $||k(p) - k^{exp}||^2$ *and the material error* $||\rho(p) - \rho^{true}||^2$ *during the iterations of the L-BFGS optimization. Although the objective function significantly decreases for the first* 50 *iterations, the material error remains close to the initial error. The same behavior is noticed in Figures B.12 and B.13.*

Piecewise-Constant	$(-100\text{nm}, 0\text{nm}) \times (-120\text{nm}, 120\text{nm})$, $(n_x = 13, n_y = 13, n_z = 2)$, $n_p = 144$
Linear	$(-100\text{nm}, 0\text{nm}) \times (-120\text{nm}, 120\text{nm})$, $(n_x = 12, n_y = 12, n_z = 1)$, $n_p = 144$
Neural Network	4 layers, $(2\times2, \text{norm.}) \to (2\times20, \tanh) \to (20\times3, \tanh) \to (3\times1, \text{id})$, $n_p = 127$

Table B.8: *Settings used for comparison of 2D parametrizations.*



(A) *Piecewise-Constant* (B) *Linear* (C) *Neural Network*

Figure B.15: *Resulting total density $\rho_{tot}(x)$ of the direct minimization of the material error $||\rho(x;p) - \rho^{true}(x)||^2$. They allow a comparison the representation capabilities of the 2D parametrizations. The reference $\rho^{true}(x)$ is the ellipsoidal inclusion from Figure B.11a. Clearly artifacts of the parametrizations are visible. The neural network performs comparatively best, but needs the most iterations and thus the longest computation time.*

error $||\rho(x;p) - \rho^{\text{true}}(x)||^2$ is optimized directly. Piecewise-constant and linear parametrization only discretize a section of the visualized domain in Figure B.11a, which is specified along with the other settings of the parametrizations in Table B.8. The neural network parametrization consists of 4 layers, with the tanh activation function applied to its intermediate layers. For a fair comparison, the number of parameters n_p of each parametrization is roughly 130.

Figure B.15 shows the results of an optimization using the L-BFGS method implemented in `Optim.jl`. Clearly, artifacts of the parametrizations are visible. The piecewise-constant parametrization cannot approximate the sharp interface of the ellipsoidal inclusion, and averages in boxes which intersect the interface. Also the linear parametrization cannot approximate the sharp interface, but represents the interface by a gradient. Both parametrizations would perform better, if the number of parameters would be higher.

Despite the slightly smaller number of parameters, the neural network approximates the ellipsoidal interface very well. However, the better approximation quality does not come without disadvantages. The number of iterations of the L-BFGS method for the minimization of the material error was

$$\text{Piecewise-Constant} : 4, \quad \text{Linear} : 23, \quad \text{NN} : 655, \tag{B.62}$$

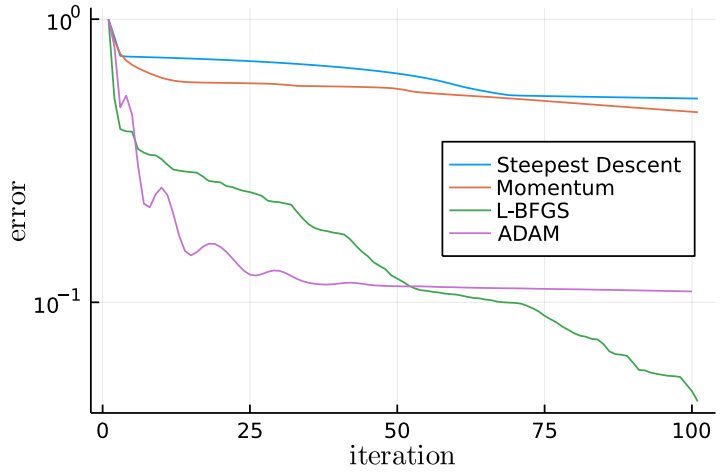


Figure B.16: *The normalized material error* $||\rho(x;p) - \rho^{true}||^2$ *during the first 100 iterations of its minimization using different optimization methods (c.f. Section B.1.2). We compare the steepest descent method, the momentum method and the L-BFGS method implemented in* `Optim.jl` *along with the ADAM method implemented in* `Flux.jl`*. Settings of all methods are left default, except for the ADAM method, where we specify* $\alpha = 0.05$.

and thus clearly the longest for the neural network. If the piecewise-constant or the linear parametrization is used, the objective function is basically quadratic in the parameters p. The L-BFGS method builds on the approximation of the objective function as a quadratic function, hence its good performance for the piecewise-constant and the linear parametrization. The neural network parametrization introduces a non-linearity which on one hand increases the representation capability, but on the other hand increases the complexity of the optimization.

Based on the optimization of the neural network parametrization, we additionally compare convergence using the different optimization methods presented in Section B.1.2. In Figure B.16 the normalized value of the objective function is visualized for the steepest descent method (blue), the momentum method (orange) and the L-BFGS method (green) which are implemented in `Optim.jl`. Additionally we visualize value of the objective minimized by the ADAM method (purple) implemented in `Flux.jl`. ADAM and L-BFGS perform significantly better for this example than steepest descent and the momentum method.

Conclusion

1 Summary

The aim of this thesis is the derivation and implementation of a reconstruction method in EPMA, which is based on a deterministic k-ratio model. Simultaneously, we target a general material description and a general objective function, which allows the extension of the method to various problem-dependent material structures and optimization methods; thus equipping the reconstruction method for a variety of questions posed to EPMA (Section 2).

In Chapter A we describe a k-ratio model based in the P_N moment expansion of the linear Boltzmann equation in Continuous Slowing Down approximation and describe the implementation of `StaRMAP.jl`, a generic solver for moment equations of a specific structure. We provide comparisons of our implementation to classical models applied for reconstruction in EPMA and showcase the flexibility of our k-ratio model.

We introduce Chapter B with the proposal to utilize gradient-based iterative optimization for the reconstruction problem in EPMA. For the application of gradient-based methods, the efficient differentiation of the forward model is crucial. Therefore, we describe a method for differentiation of the presented forward model. Our differentiation method combines the adjoint state method with the adjoint mode of algorithmic differentiation and can be implemented as an enclosed module for usage with multiple algorithmic differentiation libraries. Thereby, the parts of the computation prior to the forward model (the material description) and the parts of the computation after the forward model (the objective function) are generic and can be handled by an AD tool.

We conclude Chapter B with a validation of our differentiation method and reconstruction experiments. By means of the different reconstruction experiments, we compare general material parametrizations. The comparison emphasizes the application of non-linear material parametrizations. In our examples, the parametrization using neural networks has proven to be flexible enough to sufficiently approximate the given materials. At the same time, however, a trade-off must be made between the higher flexibility and the more complex optimization as compared to linear parametrizations.

The example in Section B.3.4 illustrates our idea of a possible real-world application of our method. The combination of measurements with prior knowledge, makes high-resolution reconstruction problems tractable with a reasonable amount of measurement data. At the same time, we demonstrate the extensibility of our implementation to various material structures.

2 High Resolution Imaging in EPMA

High resolution imaging, i.e. the reconstruction of material structures which are smaller than the interaction volume, is possible with sufficient data or sufficient material assumptions. But there will always be a relation between the accuracy of the measurement and the model to the accuracy of the reconstruction. Especially with respect to high-resolution, the analysis of this relation is key. Due to the counting in detecting x-rays, the intensity measurements in EPMA are governed by a Poisson counting process; one of many sources of measurement uncertainty in EPMA. Simultaneously, also model parameters e.g. the beam position or size and material parameters e.g. the stopping power are uncertain. To achieve a reliable reconstruction result, the quantification and propagation of uncertainties in the whole reconstruction process is necessary.

While writing this thesis, we discovered the possibility of implementing an optimization method that is based on different approximations of the gradient. The forward model consists of the composition of multiple operators, which depend on the material parameters. However, the complexity and runtime to compute the derivative of each operator differs. A reconstruction method, that builds on an approximate gradient (which neglects the dependency of some model operators) and adapts the approximation with increasing iterations, can reduce the runtime of a reconstruction.

In addition to the P_N-equations, Bünger [2021] also discusses the filtered P_N-equations which provide similar approximation quality with a smaller number of moments. Further research on the forward model could additionally include adaptability in the spatial discretization and in the number of moments or specialization on specific material parametrizations. The runtime of the forward model always determines the runtime of the reconstruction, hence any reduction is useful.

In general, we encourage further research on reconstruction in EPMA using deterministic transport equations. Although the combination with experimental data will probably raise many challenges, the presented results convince us that high-resolution imaging in EPMA is possible.

Bibliography

Benning, M. and Burger, M. (2018). Modern Regularization Methods for Inverse Problems. *arXiv:1801.09922 [math]*.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Information science and statistics. Springer, New York.

Bote, D. and Salvat, F. (2008). Calculations of inner-shell ionization by electron impact with the distorted-wave and plane-wave Born approximations. *Physical Review A*, 77(4):042701. Publisher: American Physical Society.

Bote, D., Salvat, F., Jablonski, A., and Powell, C. J. (2009). Cross sections for ionization of K, L and M shells of atoms by impact of electrons and positrons with energies up to 1GeV: Analytical formulas. *Atomic Data and Nuclear Data Tables*, 95(6):871–909.

Buenger, J., Richter, S., and Torrilhon, M. (2021). A Model for Characteristic X-Ray Emission in Electron Probe Microanalysis based on the Spherical Harmonic (PN) Method for Electron Transport. *Preprints*.

Buse, B. and Kearns, S. L. (2020). Spatial resolution limits of EPMA. *IOP Conference Series: Materials Science and Engineering*, 891:012005.

Bünger, J. (2021). *Three-dimensional modelling of x-ray emission in electron probe microanalysis based on deterministic transport equations*. PhD thesis, RWTH Aachen University. Number: RWTH-2021-05180.

Bünger, J., Sarna, N., and Torrilhon, M. (2021). Stable Boundary Conditions and Discretization for PN Equations. *arXiv:2004.02497 [cs, math]*.

Carpenter, P. K. and Jolliff, B. L. (2015). Improvements in EPMA: Spatial Resolution and Analytical Accuracy. *Microscopy and Microanalysis*, 21(S3):1443–1444. Publisher: Cambridge University Press.

Cercignani, C. (1988). *The Boltzmann Equation and Its Applications*. Applied Mathematical Sciences. Springer-Verlag, New York.

Chantler, C., Olsen, K., Dragoset, R., Chang, J., Kishore, A., Kotochigova, S., and Zucker, D. (2009). X-Ray Form Factor, Attenuation, and Scattering Tables. NIST Standard Reference Database 66.

Claus, B. E. H., Jin, Y., Gjesteby, L. A., and Wang, G. (2017). Metal-Artifact Reduction Using Deep-Learning Based Sinogram Completion: Initial Results. In *Fully3D 2017 Proceedings*, pages 631–635, Xi'an.

Claus, T., Bünger, J., and Torrilhon, M. (2021). A Novel Reconstruction Method to Increase Spatial Resolution in Electron Probe Microanalysis. *Mathematical and Computational Applications*, 26(3):51. Publisher: Multidisciplinary Digital Publishing Institute.

Cullen, D. E., Hubbell, J. H., and Kissel, L. (1997). EPDL97: the evaluated photo data library '97 version. Technical Report UCRL-50400-Vol.6-Rev.5, Lawrence Livermore National Lab. (LLNL), Livermore, CA (United States).

Duclous, R., Dubroca, B., and Frank, M. (2010). A deterministic partial differential equation model for dose calculation in electron radiotherapy. *Physics in Medicine and Biology*, 55(13):3843–3857.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

Griewank, A. (2003). A mathematical view of automatic differentiation. *Acta Numerica*, 12:321–398. Publisher: Cambridge University Press.

Hager, W. W. and Zhang, H. (2005). A New Conjugate Gradient Method with Guaranteed Descent and an Efficient Line Search. *SIAM Journal on Optimization*, 16(1):170–192. Publisher: Society for Industrial and Applied Mathematics.

Heinrich, K. F. J. and Newbury, D., editors (1991). *Electron Probe Quantitation*. Springer US.

Innes, M. (2018a). Don't Unroll Adjoint: Differentiating SSA-Form Programs. *CoRR*, abs/1810.07951.

Innes, M. (2018b). Flux: Elegant Machine Learning with Julia. *Journal of Open Source Software*.

Iserles, A. (1999). *Acta Numerica 1999: Volume 8*. Cambridge University Press.

Kelley, C. T. (2021). ReverseDiff. GitHub Repository, https://github.com/JuliaDiff/ReverseDiff.jl.

Kingma, D. P. and Ba, J. (2017). Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*.

Larsen, E. W., Miften, M. M., Fraass, B. A., and Bruinvis, I. A. (1997). Electron dose calculations using the Method of Moments. *Medical Physics*, 24(1):111–125.

Leshno, M., Lin, V. Y., Pinkus, A., and Schocken, S. (1993). Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, 6(6):861–867.

Llovet, X. and Merlet, C. (2010). Electron Probe Microanalysis of Thin Films and Multilayers Using the Computer Program XFILM. *Microscopy and Microanalysis*, 16:21–32. Publisher: Cambridge University Press (CUP).

Llovet, X., Moy, A., Pinard, P. T., and Fournelle, J. H. (2021). Electron probe microanalysis: A review of recent developments and applications in materials science and engineering. *Progress in Materials Science*, 116:100673.

Llovet, X. and Salvat, F. (2017). PENEPMA: A Monte Carlo Program for the Simulation of X-Ray Emission in Electron Probe Microanalysis. *Microscopy and Microanalysis*, 23(3):634–646. Publisher: Cambridge Univ Press.

Ma, Y., Dixit, V., Innes, M., Guo, X., and Rackauckas, C. (2021). A Comparison of Automatic Differentiation and Continuous Sensitivity Analysis for Derivatives of Differential Equation Solutions. *arXiv:1812.01892 [cs]*.

Mevenkamp, N. (2016). Inverse modeling in electron probe microanalysis based on deterministic transport equations. *Masterarbeit, RWTH Aachen, 2013*.

Mogensen, P. K. and Riseth, A. N. (2018). Optim: A mathematical optimization package for Julia. *Journal of Open Source Software*, 3(24):615.

Moy, A. and Fournelle, J. (2017). Analytical Spatial Resolution in EPMA: What is it and How can it be Estimated? *Microscopy and Microanalysis*, 23(S1):1098–1099. Publisher: Cambridge University Press.

Moy, A., Fournelle, J. H., and von der Handt, A. (2019). Solving the iron quantification problem in low-kV EPMA: An essential step toward improved analytical spatial resolution in electron probe microanalysis—Olivines. *American Mineralogist*, 104(8):1131–1142.

Müller, C. (1966). *Spherical Harmonics*. Lecture Notes in Mathematics. Springer-Verlag, Berlin Heidelberg.

Naumann, U. (2011). *The Art of Differentiating Computer Programs*. Software, Environments and Tools. Society for Industrial and Applied Mathematics.

Nocedal, J. and Wright, S. (2006). *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. Springer-Verlag, New York, 2 edition.

Pinard, P. T. (2016). *Electron probe microanalysis of carbon containing steels at a high spatial resolution*. PhD thesis, RWTH Aachen University, Aachen.

Pinard, P. T., Schwedt, A., Ramazani, A., Prahl, U., and Richter, S. (2013). Characterization of Dual-Phase Steel Microstructure by Combined Submicrometer EBSD and EPMA Carbon Measurements. *Microscopy and Microanalysis*, 19(4):996–1006. Publisher: Cambridge University Press.

Plessix, R.-E. (2006). A review of the adjoint-state method for computing the gradient of a functional with geophysical applications. *Geophysical Journal International*, 167(2):495–503.

Pouchou, J.-L. and Pichoir, F. (1991). Quantitative Analysis of Homogeneous or Stratified Microvolumes Applying the Model "PAP". In Heinrich, K. F. J. and Newbury, D. E., editors, *Electron Probe Quantitation*, pages 31–75. Springer US, Boston, MA.

Reimer, L. (1998). *Scanning Electron Microscopy: Physics of Image Formation and Microanalysis*. Springer Series in Optical Sciences. Springer-Verlag, Berlin Heidelberg, 2 edition.

Revels, J., Lubin, M., and Papamarkou, T. (2016). Forward-Mode Automatic Differentiation in Julia. *arXiv:1607.07892 [cs]*.

Richter, S., Pinard, P., Mevenkamp, N., and Torrilhon, M. (2013). High-Resolution Quantification Across Vertical Interfaces using a Monte Carlo Based Reconstruction Approach. *Microscopy and Microanalysis*, 19:1296–1297.

Ritchie, N. (2020). BoteSalvatICX.jl. GitHub Repository, https://github.com/usnistgov/BoteSalvatICX.jl.

Ritchie, N. (2021a). FFAST.jl. GitHub Repository, https://github.com/usnistgov/FFAST.jl.

Ritchie, N. (2021b). NeXLCore.jl. GitHub Repository, https://github.com/usnistgov/NeXLCore.jl.

Ritchie, N. W. M. (2009). Spectrum Simulation in DTSA-II. *Microscopy and Microanalysis*, 15(5):454–468. Publisher: Cambridge Univ Press.

Riveros, J. and Castellano, G. (1993). Review of phi(rhoz) curves in electron probe microanalysis. *X-Ray Spectrometry*, 22(1):3–10.

Salvat, F., Jablonski, A., and Powell, C. J. (2005). ELSEPA—Dirac partial-wave calculation of elastic scattering of electrons and positrons by atoms, positive ions and molecules. *Computer Physics Communications*, 165:157–190.

Seibold, B. and Frank, M. (2014). StaRMAP - A second order staggered grid method for spherical harmonics moment equations of radiative transfer. *arXiv:1211.2205 [math-ph, physics:physics]*.

Stuart, A. M. (2010). Inverse problems: A Bayesian perspective. *Acta Numerica*, 19:451–559. Publisher: Cambridge University Press.

Tarantola, A. (2005). *Inverse Problem Theory and Methods for Model Parameter Estimation*. Other Titles in Applied Mathematics. Society for Industrial and Applied Mathematics.

Thwaites, D. I. (1983). Bragg's Rule of Stopping Power Additivity: A Compilation and Summary of Results. *Radiation Research*, 95(3):495–518. Publisher: Radiation Research Society.

Appendix

C.1 Explicit Formulas for the Boundary Matrix

Here we present formulas for the P_N boundary matrices $L^{(n)}$ which are derived in Bünger [2021]. These explicit formulas can directly be implemented for two spherical harmonics (l,k) and (l',k'). The coefficient C_l^k is the normalization constant defined in Equation (A.14), $n!!$ is the double factorial, the product of all integers from 1 to n that have the same parity as n, $\Gamma(n) = (n-1)!$ is the Gamma function and $\binom{\cdot}{\cdot}$ is the binomial coefficient.

$$
\begin{aligned}
L^{(x)}_{(l,k),(l',k')} =& C_l^{|k|} C_{l'}^{|k'|} (1+(-1)^{l+l'})(-1)^{|k|+|k'|} \sum_{j=0}^{\lfloor \frac{l-|k|}{2} \rfloor} \sum_{q=0}^{\lfloor \frac{l'-|k'|}{2} \rfloor} \sum_{i=0}^{\lfloor \frac{|k|-1}{2} \rfloor} \sum_{s=0}^{\lfloor \frac{|k'|-1}{2} \rfloor} \\
& \frac{(-1)^{q+j}}{2^{j+q} j! q!} \frac{(2(l-j)-1)!!}{(l-|k|-2j)!} \frac{(2(l'-q)-1)!!}{(l'-|k'|-2q)!} \frac{\Gamma(\frac{l+l'+1-2(q+j)-(|k|+|k'|)}{2})}{\Gamma(\frac{l+l'}{2}+1-(q+j))} \\
& (-1)^{s+i} \begin{cases} \binom{k}{2i} \binom{k'}{2s} \Gamma(\frac{1}{2}+s+i) \Gamma(\frac{k+k'}{2}-(s+i)) & k,k' \geq 0 \text{ odd} \\ \binom{|k|}{2i+1} \binom{|k'|}{2s+1} \Gamma(\frac{3}{2}+s+i) \Gamma(\frac{|k|+|k'|}{2}-(s+i)-1) & k,k' < 0 \text{ even} \end{cases}
\end{aligned} \tag{C.1a}
$$

$$
\begin{aligned}
L^{(y)}_{(l,k),(l',k')} =& C_l^{|k|} C_{l'}^{|k'|} (1+(-1)^{l+l'})(-1)^{|k|+|k'|} \sum_{j=0}^{\lfloor \frac{l-|k|}{2} \rfloor} \sum_{q=0}^{\lfloor \frac{l'-|k'|}{2} \rfloor} \sum_{i=0}^{\lfloor \frac{|k|-1}{2} \rfloor} \sum_{s=0}^{\lfloor \frac{|k'|-1}{2} \rfloor} \\
& \frac{(-1)^{q+j}}{2^{j+q} j! q!} \frac{(2(l-j)-1)!!}{(l-|k|-2j)!} \frac{(2(l'-q)-1)!!}{(l'-|k'|-2q)!} \frac{\Gamma(\frac{l+l'+1-2(q+j)-(|k|+|k'|)}{2})}{\Gamma(\frac{l+l'}{2}+1-(q+j))} \\
& (-1)^{s+i} (s+i)! \binom{|k|}{2i+1} \binom{|k'|}{2s+1} \Gamma(\frac{|k|+|k'|}{2}-s-i-\frac{1}{2})
\end{aligned} \tag{C.1b}
$$

$$
\begin{aligned}
L^{(z)}_{(l,k),(l',k')} =& \pi C_l^{|k|} C_{l'}^{|k'|} |k|! \delta_{k,k'} \sum_{j=0}^{\lfloor \frac{l-|k|}{2} \rfloor} \sum_{q=0}^{\lfloor \frac{l'-|k'|}{2} \rfloor} \\
& \frac{1}{(-2)^{q+j}} \frac{1}{q! j!} \frac{(2(l-j)-1)!!}{(l-|k|-2j)!} \frac{(2(l'-q)-1)!!}{(l'-|k|-2q)!} \frac{\Gamma(\frac{l+l'}{2}-q-j-|k|)}{\Gamma(\frac{l+l'}{2}+1-q-j)}
\end{aligned} \tag{C.1c}
$$

C.2 Implementation of an Ellipsoidal Material Structure into the Existing Code

We describe the additional steps to implement a 2D ellipsoidal material structure into the existing code as a custom layer of the neural network parametrization (Section A.3.3.3). The reconstruction shown in Section B.3.4 is based on this implementation.

In 2D an ellipse is defined by its center (μ_1, μ_2), a rotation r and scaling factors in its principal axis a and b. We define the custom layer (with $(x_1, x_2)^T \in \mathbb{R}^2$ as inputs and $z \in \mathbb{R}$ as output)

$$z = \frac{((x_1 - \mu_1)\cos(r) + (x_2 - \mu_2)\sin(r))^2}{a^2} + \frac{((x_1 - \mu_1)\sin(r) - (x_2 - \mu_2)\cos(r))^2}{b^2}. \tag{C.2}$$

For completeness, we additionally derive the adjoint version of Equation (C.2). This task could just as well be done by an AD tool. We define two intermediate variables that occur repeatedly in Equation (C.2).

$$\begin{aligned} d_1 &= x_1 - \mu_1 \\ d_2 &= x_2 - \mu_2 \end{aligned} \tag{C.3}$$

Then the adjoint version can be derived from partial derivatives of Equation (C.2).

$$\begin{aligned} \bar{d_1} &= \left(\frac{2\cos(r)(d_1\cos(r) + d_2\sin(r))}{a^2} + \frac{2\sin(r)(d_1\sin(r) - d_2\cos(r))}{b^2}\right)\bar{z} \\ \bar{d_2} &= \left(\frac{2\sin(r)(d_1\cos(r) + d_2\sin(r))}{a^2} - \frac{2\cos(r)(d_1\sin(r) - d_2\cos(r))}{b^2}\right)\bar{z} \\ \bar{r} &= \frac{2(a^2 - b^2)(d_1\cos(r) + d_2\sin(r))(d_1\sin(r) - d_2\cos(r))}{a^2 b^2}\bar{z} \\ \bar{a} &= \frac{-2(d_1\cos(r) + d_2\sin(r))^2}{a^3}\bar{z} \\ \bar{b} &= \frac{-2(d_1\sin(r) - d_2\cos(r))^2}{b^3}\bar{z} \\ \bar{\mu_1} &= -\bar{d_1} \\ \bar{\mu_2} &= -\bar{d_2} \\ \bar{x_1} &= \bar{d_1} \\ \bar{x_2} &= \bar{d_2} \end{aligned} \tag{C.4}$$

The parametrization, which is applied in the reconstruction example in Section B.3.4 consists of a normalization layer, the described ellipsoidal layer and a 1×1 dense output layer with a sigmoid activation function. The output layer models the interface structure (sharp vs diffusive) of the ellipsoidal inclusion in the material. The output of the sigmoid, which by definition is $\in [0, 1]$ specifies the volume fraction of Cu, the volume fraction of Fe is deduced from the constraint $\varphi_{Cu} + \varphi_{Fe} = 1$.

The specification of the ellipse layer is related to the concept of feature extraction in machine learning [Bishop, 2006]. Feature extraction is usually applied to reduce the dimensionality of the data and to accelerate the optimization. Normally, no new parameters are introduced for feature extraction layers. For our use case, the ellipsoidal layer facilitates the interpretation of the subsequent output layer as the interface structure between inclusion and substrate.

In similar manner many other material structures can be implemented. Note that for the implementation of adjoints only the respective forward evaluation is analyzed. Afterwards the parametrization can seamlessly be integrated into the existing implementation.