UNIVERSITY OF MANNHEIM

MASTER THESIS

Quasi-Monte Carlo Methods and Applications

Author:
Janik V. HRUBANT

Supervisors:
Prof. Dr. Andreas
NEUENKIRCH

A thesis submitted in fulfillment of the requirements for the degree of Master of Science

in the

Research Group Name
Department or School Name

August 7, 2025

Declaration of Authorship

English Version. I hereby declare that I have completed this thesis independently and without any unauthorized assistance. I further confirm that neither this thesis nor any part of it has been submitted by me or by others as part of any other academic assessment. Literal or paraphrased quotations from other sources—whether in print or electronic form—are clearly marked as such. All secondary literature and other sources used are properly cited and listed in the bibliography. This also applies to graphical representations, images, and all online sources. I furthermore agree that my thesis may be submitted and stored in anonymized electronic form for the purpose of plagiarism detection. I am aware that the thesis may not be evaluated if this declaration is not submitted.

German Version. Hiermit versichere ich, dass diese Arbeit von mir persönlich verfasst wurde und dass ich keinerlei fremde Hilfe in Anspruch genommen habe. Ebenso versichere ich, dass diese Arbeit oder Teile daraus weder von mir selbst noch von anderen als Leistungsnachweise andernorts eingereicht wurden. Wörtliche oder sinngemäße Übernahmen aus anderen Schriften und Veröffentlichungen in gedruckter oder elektronischer Form sind gekennzeichnet. Sämtliche Sekundärliteratur und sonstige Quellen sind nachgewiesen und in der Bibliographie aufgeführt. Das Gleiche gilt für graphische Darstellungen und Bilder sowie für alle Internet-Quellen. Ich bin ferner damit einverstanden, dass meine Arbeit zum Zwecke eines Plagiatsabgleichs in elektronischer Form anonymisiert versendet und gespeichert werden kann. Mir ist bekannt, dass von der Korrektur der Arbeit abgesehen werden kann, wenn diese Erklärung nicht erteilt wird.

Place, Date

Signature

Abstract

Here comes the Abstract or Introduction of the thesis.

Contents

RITA Rational Inverse Transform with Aliasing

CDF Cumulative Distribution Function

QMC Quasi-Monte Carlo

MC Monte Carlo

HU Hounsfield Unit

CT Computed Tomography

CBCT Cone-Beam Computed Tomography

FFD Forced Fixed Detection

CNN Convolutional Neural Network

DNN Deep Neural Network

PDE Partial Differential Equation

Part I

Fundamentals of Quasi-Monte Carlo Methods

Chapter 1

Monte Carlo and Quasi-Monte Carlo Integration

1.1 Motivation and Problem Setting

The numerical evaluation of high-dimensional integrals is a fundamental task in modern applied mathematics and scientific computing. Applications range from Bayesian inference and financial mathematics to machine learning and computational physics. In particular, contemporary domains such as deep neural network training and medical imaging often require the estimation of integrals of the form

$$I(f) = \int_{[0,1]^s} f(x)\, dx\,, \tag{1.1}$$

where s denotes the dimensionality of the problem and f is a function that may be expensive or impractical to evaluate analytically.

One of the most widely used approaches to compute such integrals is the classical Monte Carlo (MC) method, which estimates the expectation based on averages over randomly sampled points. The primary appeal of MC integration lies in its dimension-independent convergence rate and minimal assumptions on the integrand. However, its asymptotic error rate of $\mathcal{O}(N^{-1/2})$ limits its efficiency – especially when high precision is required or function evaluations are computationally expensive.

Quasi-Monte Carlo (QMC) methods offer an alternative paradigm: instead of random samples, they employ deterministic sequences – so-called low-discrepancy sequences—that aim to fill the integration domain more uniformly. This structured sampling allows for faster convergence under certain smoothness conditions and forms the basis for many state-of-the-art techniques in high-dimensional numerical integration.

The goal of this part is to develop a rigorous mathematical foundation for QMC methods and to understand how they improve upon classical MC integration. The key concepts introduced here – particularly discrepancy theory and low-discrepancy sequences – serve as theoretical tools that will be revisited in later parts of this thesis. Chapter 2 will delve into the measurement of uniformity via star discrepancy and provide concrete constructions of low-discrepancy sequences such as Sobol' and Halton, which are central to the numerical methods applied in Part II and Part III, dedicated to neural network training and CT-based photon transport simulation.

1.1.1 High-Dimensional Integration in Applications

High-dimensional integration problems arise naturally in a wide range of scientific and engineering disciplines. Whenever expectations with respect to multivariate distributions must be computed numerically, they are typically formulated as integrals over the s-dimensional unit cube – such as the integral in Equation (1.1).

Prominent examples include:

- **Bayesian statistics**: Computing posterior expectations, marginal likelihoods, or predictive distributions.
- **Financial mathematics**: Pricing complex financial derivatives and evaluating risk measures under stochastic models.
- **Machine learning**: Estimating expectations in variational inference, training neural networks using randomized optimization techniques, or evaluating generalization bounds.
- **Medical imaging**: Simulating photon transport and estimating physical quantities based on noisy measurements, especially in Computed Tomography (CT) and magnetic resonance imaging (MRI).

In all these cases, the dimensionality s can be moderate to very high – sometimes even exceeding hundreds or thousands of variables. This introduces significant challenges for numerical integration methods, which must balance accuracy, computational cost, and robustness with respect to the structure of the integrand.

The remainder of this chapter explores how MC and QMC methods address these challenges. Before doing so, we briefly review the MC method and its fundamental convergence properties in the next section.

1.1.2 Monte Carlo Integration: Principle and Convergence

The classical MC method is a probabilistic approach to numerical integration that relies on random sampling. It is particularly suited for high-dimensional settings, as its convergence rate does not deteriorate with increasing dimension.

Definition 1.1.1 (Monte Carlo Estimator)**.**
Let $f \in L^2([0,1]^s)$ and let $X_0, \dots, X_{N-1} \sim \mathcal{U}([0,1]^s)$ be independent and identically distributed random samples. The Monte Carlo estimator of the integral I is given by

$$I_N^{\mathrm{MC}}(f) := \frac{1}{N} \sum_{n=0}^{N-1} f(X_n)\,. \tag{1.2}$$

This estimator is unbiased and converges almost surely to the true integral as $N \to \infty$, as established by the Strong Law of Large Numbers:

Theorem 1.1.2 (Strong Law of Large Numbers)**.**
Let $f \in L^2([0,1]^s)$. Then

$$\mathbb{P}\left[\lim_{N\to\infty} I_N^{\mathrm{MC}}(f) = \int_{[0,1]^s} f(x)\,dx\right] = 1. \tag{1.3}$$

Beyond this almost sure convergence, the expected integration error of the Monte Carlo estimator can be quantified using the root-mean-square error [15, Section 1.3]:

Theorem 1.1.3 (Monte Carlo Convergence Rate)**.**
Let $f \in L^2([0,1]^s)$. Then

$$\mathbb{E}\left[\left|I_N^{\mathrm{MC}}(f) - I(f)\right|\right] \leq \frac{\sigma[f]}{\sqrt{N}}, \tag{1.4}$$

where $\sigma[f] := \sqrt{\mathrm{Var}[f]}$.

Sketch of Proof. Using independence and linearity of expectation, the variance of $I_N^{\mathrm{MC}}(f)$ is

$$\mathrm{Var}[I_N^{\mathrm{MC}}(f)] = \frac{\mathrm{Var}[f]}{N}. \tag{1.5}$$

Applying Jensen's inequality yields the stated bound on the expected absolute error. □

Remark 1.1.4. The convergence rate $\mathcal{O}(N^{-1/2})$ is independent of the integration dimension s, which is a key advantage of the MC method. In contrast, classical grid-based methods often suffer from the curse of dimensionality, where the number of required samples grows exponentially with s.

Example 1.1.5. Let $f \in C^1([0,1]^s)$ be a Lipschitz-continuous function. A uniform grid with $N = m^s$ points has a worst-case error of $\mathcal{O}(N^{-1/s})$. In high dimensions, this becomes prohibitively inefficient, while MC integration retains the dimension-agnostic rate $\mathcal{O}(N^{-1/2})$. [6, Section 1.1]

Despite its robustness and simplicity, MC integration has several well-known limitations:

- The convergence rate is relatively slow, especially for smooth integrands.
- Error bounds are probabilistic rather than deterministic.
- The method does not exploit structural properties of the integrand, such as smoothness or sparsity.

These limitations motivate the development of QMC methods, which will be introduced in the next section.

1.2 Quasi-Monte Carlo Methods

1.2.1 Deterministic Sampling and Intuition

Classical MC methods approximate integrals over the unit cube $[0,1]^s$ using randomly sampled points. In contrast, QMC methods replace stochasticity with a deterministic strategy, aiming to cover the integration domain in a more uniform and structured way.

Definition 1.2.1 (Quasi-Monte Carlo Estimator).
Let $f\colon [0,1]^s \to \mathbb{R}$ be a measurable function and let $\{\boldsymbol{x}_n\}_{n=1}^N \subset [0,1]^s$ be a deterministic point set. Then the *quasi-Monte Carlo estimator* for the integral

$$I(f) = \int_{[0,1]^s} f(\boldsymbol{x})\, d\boldsymbol{x}$$

is defined as

$$I_N^{\mathrm{QMC}}(f) := \frac{1}{N} \sum_{n=1}^{N} f(\boldsymbol{x}_n). \tag{1.6}$$

The functions I, I_N^{MC} and I_N^{QMC} will be further used without explicit reference to the integrand f when the context is clear.

Unlike in the Monte Carlo setting, these points are not drawn from a probability distribution, but are generated deterministically — typically by rules that aim to avoid clustering and oversampling.

Figure 1.1 illustrates this contrast by comparing 300 randomly sampled points to 300 QMC points from the Sobol' sequence in two dimensions. The QMC points distribute more evenly, avoiding both gaps and clusters.

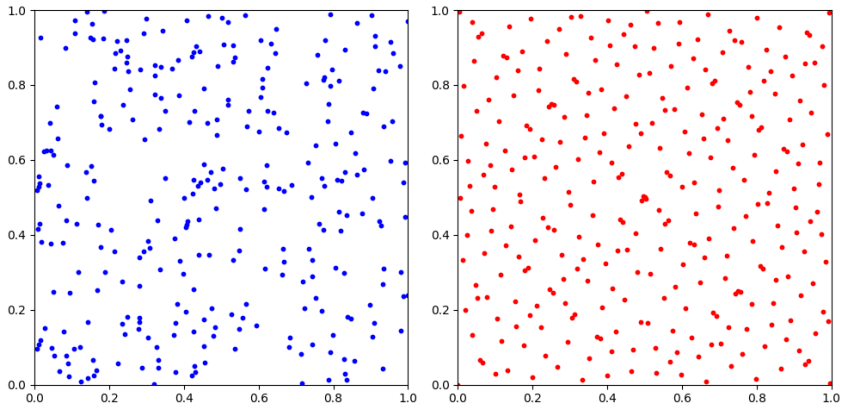


FIGURE 1.1: Comparison of 300 sample points in $[0,1]^2$ using (left) standard MC and (right) a Sobol' sequence. QMC points avoid clustering and fill the domain more evenly.

A natural alternative to MC sampling is the use of regular tensor-product grids. However, these grids suffer from the *curse of dimensionality*: placing m points per dimension leads to a total of $N = m^s$ points, which becomes infeasible even for moderate s. Furthermore, grids are not easily extensible – adding new points requires

global recomputation and destroys nesting.

By contrast, many QMC sequences, such as Sobol' and Halton (will be introduced in Chapter 2), are designed to be *incremental*: each new point can be added without changing the existing ones. This makes QMC particularly suitable for adaptive algorithms, especially for anytime algorithms that might increment the number of samples needed over time. However, not all QMC constructions share this feature such as lattice rules and some optimized designs are fixed-size by nature.

QMC methods thus offer the best of both worlds: they combine the space-filling structure of grids with the flexibility and scalability of sampling methods. Low-discrepancy sequences fill the space more evenly than random points while avoiding the redundancy of regular grids. This often leads to significantly lower integration errors, especially for smooth functions or problems with low effective dimension.

This shift from probabilistic to deterministic sampling also changes the way we analyze error: instead of using statistical bounds, we rely on *discrepancy measures*, which quantify the uniformity of the point set. These concepts, along with the notion of function variation, will be introduced in the following sections.

Remark 1.2.2. QMC estimators retain the same algebraic structure as their MC counterparts, but their convergence behavior is governed by entirely different theoretical tools – namely discrepancy theory and function variation.

1.2.2 Monte Carlo vs. Quasi-Monte Carlo: A First Comparison

MC and QMC methods share the same high-level goal: estimating an integral by averaging function evaluations at selected sample points. The difference lies in the sampling strategy – stochastic versus deterministic – and the consequences this has for accuracy, convergence and theoretical guarantees.

Table 1.1 summarizes key conceptual distinctions between the two paradigms.

Aspect	**Monte Carlo (MC)**	**Quasi-Monte Carlo (QMC)**
Sampling	Independent random points	Deterministic low-discrepancy points
Error bounds	Probabilistic (in expectation)	Deterministic (worst-case)
Regularity assumptions on f	Square integrability (L^2)	Bounded variation (HK)
Theoretical convergence	$\mathcal{O}(N^{-1/2})$	Up to $\mathcal{O}(N^{-1})$ (heuristic)
Point extensibility	Trivial	Often supported (e.g., Sobol)
Robustness to noise	High	Low

TABLE 1.1: Conceptual comparison of MC and QMC integration methods.

While MC estimators are unbiased and robust even under minimal assumptions, their convergence is slow and does not improve when the integrand is smooth. QMC methods, on the other hand, exploit structure in the integrand – such as smoothness or low effective dimension – and can yield significantly lower integration errors.

However, QMC methods lack probabilistic guarantees and depend more strongly on the careful construction of the point set. Their performance can degrade if the integrand exhibits high variability along many input dimensions or if the chosen sequence is not well matched to the function.

The next subsection discusses the convergence rates of both methods in more detail and highlights the interplay between dimensionality and integration error.

1.3 Uniformity Concepts

The efficiency of QMC methods hinges on how well the employed point sets "fill" the integration domain $[0,1]^s$. This motivates the need for a precise mathematical understanding of *uniformity*. The present section introduces two key concepts in this regard – *uniform distribution modulo one* and *equidistribution* – and clarifies their relevance to numerical integration.

1.3.1 Uniform Distribution vs. Equidistribution

While the terms *uniform distribution* and *equidistribution* are sometimes used interchangeably in informal contexts, they carry distinct meanings in the context of numerical integration. In QMC methods, the notion of *uniform distribution modulo one* provides a rigorous criterion for how evenly a sequence covers the unit cube. This concept forms the foundation for analyzing the convergence behavior of QMC estimators.

Definition 1.3.1 (Uniform Distribution Modulo One)**.**
Let $(\boldsymbol{x}_n)_{n\in\mathbb{N}_0} \subset [0,1]^s$ be a sequence of sample points. The sequence is said to be *uniformly distributed modulo one* (u.d. mod 1) if for every axis-aligned box $[\boldsymbol{ab}) \subset [0,1]^s$, the proportion of points falling into this box converges to its Lebesgue measure:

$$\lim_{N\to\infty} \frac{1}{N} \sum_{n=0}^{N-1} \chi_{[\boldsymbol{a},\boldsymbol{b})}(\boldsymbol{x}_n) = \lambda_s([\boldsymbol{a},\boldsymbol{b}))\,,$$

where χ_A is the indicator function of the set A and λ_s denotes the s-dimensional Lebesgue measure.

This definition emphasizes asymptotic spatial coverage: in the limit, every subregion of the domain is sampled proportionally to its volume. Importantly, this property is purely deterministic and does not rely on any probabilistic assumptions — in contrast to Monte Carlo methods, which only guarantee uniformity in expectation.

In contrast, when speaking of a random sample $\{X_1,\dots,X_N\} \subset [0,1]^s$ drawn independent and identically distributed (i.i.d.) from the uniform distribution, we refer to a probabilistic concept of uniformity: each point is independently drawn according to the uniform measure, but the empirical distribution may not be uniformly spread in finite samples. Thus, equidistribution refers to the ideal uniform coverage that we seek deterministically, while uniform random sampling only ensures this behavior in expectation or with high probability.

Remark 1.3.2. Uniform distribution modulo one is a necessary condition for the convergence of QMC estimators. If a point sequence is not u.d. mod 1, then there exists at least one Riemann-integrable function for which the sample average fails to converge to the integral.

The following result, known as Weyl's criterion, provides a necessary and sufficient condition for uniform distribution in terms of exponential sums. A proof can be found in [6].

Theorem 1.3.3 (Weyl's Criterion)**.**
A sequence $(\boldsymbol{x}_n)_{n\geq 0} \subset [0,1)^s$ is uniformly distributed modulo one if and only if, for all nonzero $\boldsymbol{h} \in \mathbb{Z}^s \setminus \{\mathbf{0}\}$, we have

$$\lim_{N\to\infty} \frac{1}{N} \sum_{n=0}^{N-1} \exp(2\pi i\, \boldsymbol{h} \cdot \boldsymbol{x}_n) = 0.$$

Example 1.3.4. Let $\alpha \in \mathbb{R}$ be irrational. Then the Kronecker sequence $(n\alpha \bmod 1)_{n\geq 0}$ is uniformly distributed in $[0,1]$. This is a consequence of Weyl's criterion and illustrates the existence of simple, deterministic sequences with excellent uniformity properties.

1.3.2 Why Uniformity Matters in Numerical Integration

In QMC integration, the objective is to approximate the integral defined in Equation (1.1) by a finite average over a deterministic point set $\{\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{N-1}\} \subset [0,1]^s$. The accuracy of this approximation depends crucially on how uniformly the point set samples the domain.

Unlike MC methods, which rely on probabilistic guarantees and variance-based error estimates, QMC methods exploit deterministic structure: the integration error is directly influenced by how well the point set fills the domain. This insight gives rise to one of the most fundamental theoretical tools in QMC analysis: the *Koksma–Hlawka inequality*.

Remark 1.3.5. The Koksma–Hlawka inequality bounds the integration error of a QMC estimator by the product of two quantities: the *star discrepancy* of the point set and the *variation* of the integrand in the sense of Hardy-Krause. In short,

$$\left| \frac{1}{N} \sum_{n=0}^{N-1} f(\boldsymbol{x}_n) - I \right| \leq D_N^*(\{\boldsymbol{x}_n\}) \cdot V_{\mathrm{HK}}(f).$$

A rigorous statement and detailed discussion of this inequality will follow in Chapter 3.

This inequality highlights why uniformity is essential: if the integrand has bounded variation, then smaller discrepancy directly implies smaller integration error. In this sense, discrepancy theory becomes a cornerstone of effective QMC integration.

Remark 1.3.6. The star discrepancy quantifies the maximal deviation between the empirical distribution of the point set and the uniform distribution over $[0,1]^s$. It vanishes asymptotically for uniformly distributed sequences and governs the convergence behavior of QMC estimators.

The upcoming Chapter 2 provides a formal introduction to discrepancy theory. We will define extreme and star discrepancy, study their geometric interpretation, and analyze the behavior of structured low-discrepancy sequences such as Halton and

Sobol'. These sequences, due to their uniform space-filling properties, play a central role in modern high-dimensional numerical integration.

Chapter 2

Discrepancy Theory and Low-Discrepancy Sequences

2.1 Measuring Uniformity: Discrepancy

One of the cornerstones of QMC theory is the concept of *discrepancy*, which quantifies how uniformly a finite point set samples the s-dimensional unit cube $[0,1]^s$. While uniform distribution modulo one describes the asymptotic behavior of infinite sequences, discrepancy provides a finite-sample measure of deviation from perfect uniformity. Low-discrepancy sequences are the foundation of QMC methods because they minimize this deviation and thus yield more accurate numerical integration.

This section introduces formal definitions, geometric intuition and empirical insights into discrepancy measures. It lays the theoretical groundwork for understanding how the structure of point sets affects QMC integration performance.

2.1.1 Definition of Discrepancy and Star Discrepancy

First, we define the more general *extreme discrepancy*:

Definition 2.1.1 (Extreme Discrepancy)**.**
Let $\mathcal{P} \subset [0,1)^s$, with $|\mathcal{P}| = N$, being a finite point set. Then the extreme discrepancy $D_N(\mathcal{P})$ is defined as

$$D_N(\mathcal{P}) = \sup_{\substack{\boldsymbol{a},\boldsymbol{b}\in[0,1]^s \\ \boldsymbol{a}\leq\boldsymbol{b}}} \left| \frac{A([\boldsymbol{a},\boldsymbol{b}),\mathcal{P},N)}{N} - \lambda_s([\boldsymbol{a},\boldsymbol{b})] \right|.$$

Hereby $A([\boldsymbol{a},\boldsymbol{b}),\mathcal{P},N)$ denotes the number of points in $\mathcal{P}$ that fall into the box $[\boldsymbol{a},\boldsymbol{b})$, and $\lambda_s([\boldsymbol{a},\boldsymbol{b})]$ is the Lebesgue measure of that box, given by $\prod_{i=1}^{s}(b_i - a_i)$.

In practice, a common and slightly more tractable variant is the *star discrepancy*, which restricts the test boxes to be anchored at the origin.

Definition 2.1.2 (Star Discrepancy)**.**
Let $\mathcal{P} \subset [0,1)^s$, with $|\mathcal{P}| = N$, being a finite point set. Then the star discrepancy D_N^* is defined as

$$D_N^*(\mathcal{P}) = \sup_{\boldsymbol{t} \in [0,1]^s} \left| \frac{A([\mathbf{0}, \boldsymbol{t}), \mathcal{P}, N)}{N} - \lambda_s([\mathbf{0}, \boldsymbol{t})] \right|.$$

This form is used in the Koksma–Hlawka inequality and serves as a central measure in QMC analysis. Note that both discrepancy definitions are deterministic and depend solely on the point configuration.

Remark 2.1.3. A low star discrepancy implies that the empirical distribution of the points approximates the uniform distribution well over all axis-aligned subrectangles anchored at the origin.

2.1.2 Geometric Interpretation and Examples

To build geometric intuition, consider the 1-dimensional case. Given N sample points $x_0, \ldots, x_{N-1} \in [0,1)$, we can visualize discrepancy as the maximum vertical deviation between the empirical distribution function

$$F_N(t) := \frac{1}{N} \sum_{n=0}^{N-1} \chi_{[0,t)}(x_n)$$

and the uniform cumulative distribution function $F(t) = t$. The discrepancy corresponds to the largest gap between $F_N(t)$ and $F(t)$.

In higher dimensions, the idea generalizes: the star discrepancy measures the maximal difference in the number of points falling into an axis-aligned box $[\mathbf{0}, \boldsymbol{t})$ versus the volume of that box. Intuitively, it quantifies whether the point set "overpopulates" or "underpopulates" certain regions.

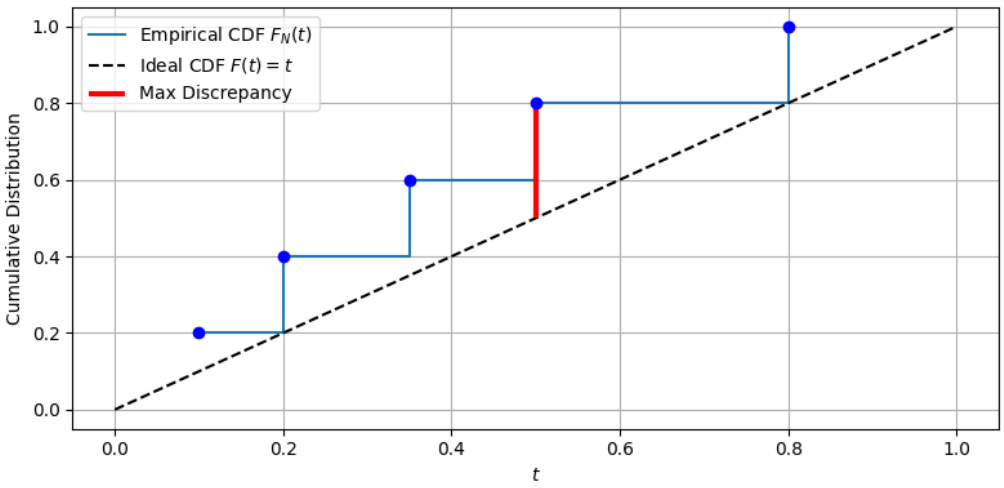


FIGURE 2.1: Visualization of 1D discrepancy: the maximum vertical distance between the empirical CDF $F_N(t)$ and the uniform CDF $F(t) = t$.

Example 2.1.4. Let $x_n = \frac{n}{N}$ for $n = 0, \ldots, N-1$. Then $F_N(t)$ is a piecewise constant staircase function, and the discrepancy can be shown to be $\mathcal{O}(1/N)$. This is an optimal rate in 1D.

Remark 2.1.5. Discrepancy provides a worst-case error metric over all subintervals. Thus, even a point set that appears visually well-distributed may have large discrepancy due to subtle gaps or clustering in certain regions.

2.1.3 Empirical Observation of Discrepancy in QMC

Discrepancy is not just a theoretical construct – its practical relevance becomes evident when comparing the behavior of QMC sequences with MC samples. In particular, structured point sets like Halton and Sobol' sequences achieve much lower discrepancy than random samples of the same size.

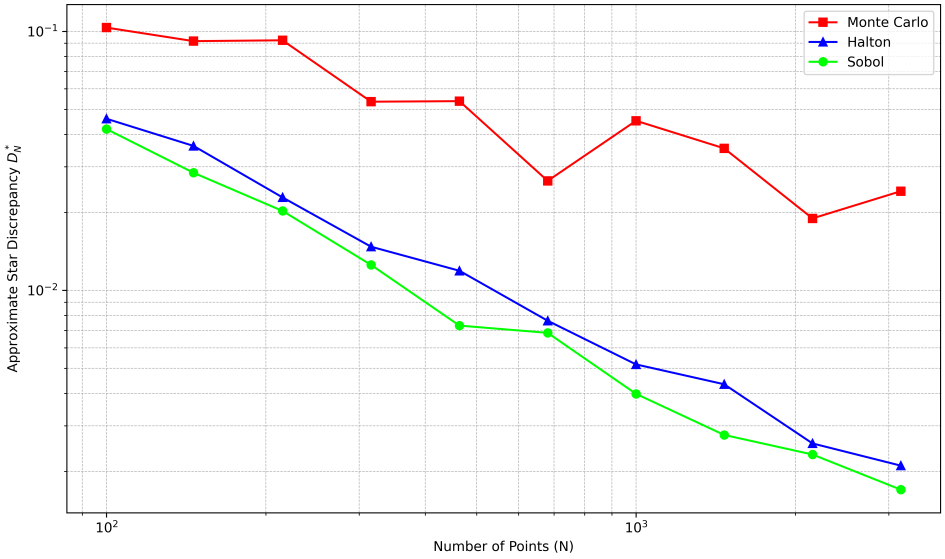


FIGURE 2.2: Comparison of discrepancy growth for Sobol', Halton and random (MC) point sets in 2D. Low-discrepancy sequences exhibit significantly slower discrepancy growth.

Remark 2.1.6. The expected star discrepancy of i.i.d. Monte Carlo samples decreases at the rate $\mathcal{O}(N^{-1/2})$, whereas for low-discrepancy sequences, provable upper bounds of order $\mathcal{O}((\log N)^s / N)$ exist. [6, Section 2.2]

The next section will introduce specific constructions of low-discrepancy sequences and analyze their dimensional performance and implementation details.

2.2 Construction of Low-Discrepancy Sequences

Before defining specific low-discrepancy sequences, we first introduce the term of low-discrepancy sequences, which are designed to fill the unit cube $[0,1)^s$ as uniformly as possible.

Definition 2.2.1 (Low-Discrepancy Sequence)**.**
A sequence $\mathcal{S} = (x_n)_{n\in\mathbb{N}}$ in $[0,1)^s$ is called a low-discrepancy sequence if its star discrepancy satisfies

$$D_N^*(\mathcal{S}) = \mathcal{O}\left(\frac{(\log N)^s}{N}\right)$$

for all $N \in \mathbb{N}$. Such sequences are also referred to as *quasi-Monte Carlo sequences* or QMC sequences.

Low discrepancy sequences are constructed to minimize the star discrepancy, which is crucial for the convergence of QMC methods according to the Koksma-Hlawka inequality introduced in Chapter 3.

2.2.1 The Halton Sequence

The Halton sequence is one of the earliest and most widely used constructions of low-discrepancy sequences in arbitrary dimensions.

For the construction of the Halton sequence, we utilize the radical-inverse functin $\phi_b(n)$. The radical-inverse function $\phi_b(n)$ maps an integer n to a real number in $[0,1)$ by reflecting its base-b representation about the decimal point:

$$\phi_b(n) = \sum_{k=0}^{\infty} d_k b^{-k-1}, \quad \text{where } n = \sum_{k=0}^{\infty} d_k b^k.$$

Definition 2.2.2 (Van der Corput Sequence)**.**
The one-dimensional van der Corput sequence in base b is defined as

$$\mathcal{S}_b = (\phi_b(n))_{n\in\mathbb{N}}\,.$$

The Halton sequence generalizes the one-dimensional van der Corput sequence to multiple dimensions by using mutually prime bases.

Definition 2.2.3 (Halton Sequence)**.**
Let $b_1, \ldots, b_s$ be pairwise coprime integers greater than 1, typically chosen as the first s prime numbers. The n-th point $\boldsymbol{x}_n \in [0,1)^s$ of the s-dimensional Halton sequence is defined componentwise by

$$\boldsymbol{x}_n = (\phi_{b_1}(n), \ldots, \phi_{b_s}(n))\,,$$

where $\phi_b(n)$ denotes the van der Corput radical-inverse function in base b. The Halton sequence is then given by $\mathcal{S}_{b_1,\ldots,b_s} = (x_n)_{n\in\mathbb{N}}$.

Example 2.2.4 (First Elements of the Halton Sequence in 2D).
Consider the two-dimensional Halton sequence based on the first two prime bases $b_1 = 2$ and $b_2 = 3$. The first three elements are obtained by computing the radical-inverse values $\phi_{b_1}(n)$ and $\phi_{b_2}(n)$ for $n = 1, 2, 3$:

- $n = 1$: $\boldsymbol{x}_1 = (\phi_2(1), \phi_3(1)) = \left(\frac{1}{2}, \frac{1}{3}\right)$

- $n = 2$: $\boldsymbol{x}_2 = (\phi_2(2), \phi_3(2)) = \left(\frac{1}{4}, \frac{2}{3}\right)$
- $n = 3$: $\boldsymbol{x}_3 = (\phi_2(3), \phi_3(3)) = \left(\frac{3}{4}, \frac{1}{9}\right)$

This illustrates how the Halton sequence fills the unit square in a low-discrepancy manner even for small n.

Intuitively, this construction ensures that each component of the sequence explores the unit interval in a structured, non-redundant way. By combining several one-dimensional van der Corput sequences in different coprime bases, the resulting multi-dimensional point set avoids regular grid-like patterns and achieves asymptotic uniformity.

As stated in [15], the Halton sequence has star discrepancy of order

$$D_N^*(\{\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{N-1}\}) = \mathcal{O}\left(\frac{(\log N)^s}{N}\right),$$

making it a prototypical example of a low-discrepancy sequence. However, for larger dimensions, correlation effects between the different base components can lead to degraded uniformity and higher discrepancy. Variants such as scrambling or leaping are commonly employed to mitigate this issue.

Remark 2.2.5. The Halton sequence is extensible in N and s, making it suitable for applications that require growing or adaptive point sets. However, its performance in high dimensions is often inferior to more modern constructions like the Sobol' sequence.

2.2.2 The Sobol' Sequence

Sobol' sequences are another class of low-discrepancy sequences that are widely used in QMC methods, particularly for high-dimensional problems. Constructed to fill the s-dimensional unit cube $[0, 1)^s$ as uniformly as possible, Sobol' sequences are based on a recursive structure that allows for efficient generation and high-dimensional performance.

Definition 2.2.6 (Sobol' Sequence Construction)**.** Let $p_1, \ldots, p_s \in \mathbb{F}_2[x]$ be primitive polynomials ordered by non-decreasing degree. Each polynomial p_j for dimension j is of the form

$$p_j(x) = x^{e_j} + a_{1,j}x^{e_j-1} + \cdots + a_{e_j-1,j}x + 1,$$

where $e_j \in \mathbb{N}$ and the coefficients $a_{k,j} \in \{0, 1\}$.
Choose initial values $m_{1,j}, \ldots, m_{e_j,j}$ such that each $m_{k,j}$ is an odd integer and $1 \leq m_{k,j} < 2^k$ for $1 \leq k \leq e_j$. For $k > e_j$, the values $m_{k,j}$ are computed recursively as:

$$m_{k,j} = a_{1,j} \cdot 2m_{k-1,j} \oplus a_{2,j} \cdot 2^2 m_{k-2,j} \oplus \cdots \oplus a_{e_j-1,j} \cdot 2^{e_j-1} m_{k-e_j+1,j} \oplus 2^{e_j} m_{k-e_j,j} \oplus m_{k-e_j,j},$$

where $\oplus$ denotes bitwise exclusive-or (XOR).
The corresponding direction numbers are defined by

$$v_{k,j} = \frac{m_{k,j}}{2^k}.$$

Let $n \in \mathbb{N}_0$ with binary expansion

$$n = n_0 + 2n_1 + \cdots + 2^{r-1}n_{r-1}, \quad n_i \in \{0, 1\}.$$

Then the j-th coordinate of the n-th Sobol' point is given by

$$x_{n,j} = n_0 v_{1,j} \oplus n_1 v_{2,j} \oplus \cdots \oplus n_{r-1} v_{r,j}.$$

The s-dimensional Sobol' point is

$$\boldsymbol{x}_n = (x_{n,1}, \ldots, x_{n,s}).$$

Remark 2.2.7. In dimension $j = 1$, the polynomial is typically chosen as $p_1(x) = x$, which leads to a construction equivalent to the van der Corput sequence in base 2 from Definition 2.2.2.

The bitwise structure of the Sobol' sequence makes it particularly efficient to generate and it enables streaming or online sampling in high dimensions.

2.2.3 Dimensional Performance and Sequence Comparison

The practical performance of low-discrepancy sequences is strongly influenced by the dimension s of the integration domain. Although both Halton and Sobol' sequences achieve the asymptotic star discrepancy bound of order $\mathcal{O}((\log N)^s / N)$ (as will be shown in Chapter 3), their behavior in higher dimensions differs significantly in practice.

Halton Sequence. While the Halton sequence performs well in low-dimensional settings, its structure leads to correlation artifacts in higher dimensions due to the use of increasing prime bases. These artifacts manifest as clustering or gaps in certain coordinate directions, which can degrade uniformity. As a result, the empirical discrepancy of the Halton sequence often grows faster than the theoretical bound suggests in moderate to high dimensions.

Sobol' Sequence. In contrast, the Sobol' sequence is explicitly constructed for high-dimensional performance. The use of carefully selected primitive polynomials and direction numbers minimizes inter-dimensional correlations. Furthermore, the bit-wise construction allows for better numerical stability and efficient implementation. Sobol' sequences generally exhibit lower discrepancy than Halton sequences in dimensions $s > 10$, making them a standard choice in modern QMC applications.

The below Figure 2.3 illustrates the difference in a two-dimensional setting between the two low discrepancy sequences. The Sobol' sequence fills the unit square more uniformly, while the Halton sequence shows visible clustering.

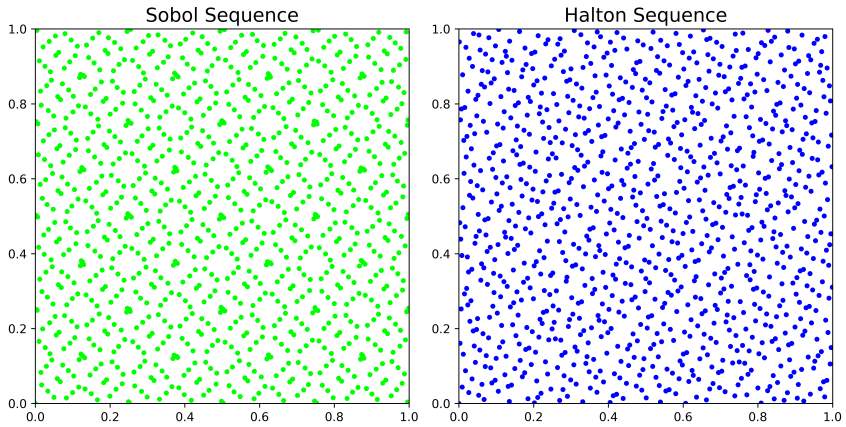


FIGURE 2.3: Comparison of Halton and Sobol' sequences in $s = 2$ dimensions for $N = 1024$ points. The Sobol' sequence fills the unit square more uniformly, demonstrating lower star discrepancy than the Halton sequence.

Remark 2.2.8. In high-dimensional problems, particularly those arising in computational finance or scientific simulations, Sobol' sequences with proper scrambling consistently outperform Halton sequences in terms of integration accuracy and numerical stability.

Summary Table. The following table summarizes the key characteristics of both sequences:

Property	**Halton**	**Sobol'**
Extensible in N	Yes	Yes
Extensible in s	Yes	Yes
Asymptotic Star Discrepancy	$\mathcal{O}\left(\frac{(\log N)^s}{N}\right)$	$\mathcal{O}\left(\frac{(\log N)^s}{N}\right)$
High-Dimensional Performance	Limited	Excellent
Implementation Cost	Low	Moderate

TABLE 2.1: Qualitative comparison of Halton and Sobol' sequences.

Chapter 3

Function Variation and Error Bounds in QMC

3.1 Function Variation in Multiple Dimensions

In Quasi-Monte Carlo (QMC) theory, the variation of a function plays a key role in bounding the integration error. While the total variation is sufficient for univariate functions, higher dimensions necessitate more refined notions. In this section, we follow the definitions of [13] and introduce two such notions: the Vitali variation and the variation in the sense of Hardy–Krause. These concepts are crucial for understanding the convergence guarantees provided by the Koksma–Hlawka inequality.

3.1.1 Mixed Component Selection and Alternating Sums

To express multidimensional variation concisely, we begin with the following notational conventions.

Definition 3.1.1 (Mixed Component Selection)**.**
Let $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^s$ and let $u \subseteq \{1, \dots, s\}$ be an index set. We define

$$\boldsymbol{c} := \boldsymbol{a}^u : \boldsymbol{b}^{-u} \quad \text{with} \quad c_j := \begin{cases} a_j, & \text{if } j \in u, \\ b_j, & \text{if } j \notin u. \end{cases}$$

Here, the term $-u$ denotes the complement of u in $\{1, \dots, s\}$. Using this notation, the *alternating sum* is defined as follows:

Definition 3.1.2 (Alternating Sum)**.**
The *alternating sum* on s dimensions with respect to a function on the hyperrectangle $[a, b]$ is defined as

$$\Delta(f; a, b) = \sum_{v \subseteq \{1,\dots,s\}} (-1)^{|v|} f(a^v : b^{-v}).$$

3.1.2 Variation in the Sense of Vitali

A classical approach to quantify function variation on hyperrectangles is based on the concept of *ladders*:

Definition 3.1.3 (Ladder)**.**
A *ladder* on the hyperrectangle $[a, b]$ is a set of points $\mathcal{Y} = \{y_1, \ldots, y_k\} \subset [a, b]$ such that for each $i < j$, the point y_i is less than or equal to y_j in every coordinate. Formally, this means that for all $i < j$ and for all $l \in \{1, \ldots, s\}$, we have $y_i^l \leq y_j^l$.

Let $\mathbb{Y}$ denote the set of all such ladders on $[a, b]$. The *variation with respect to a ladder* is then defined as:

Definition 3.1.4 (Variation with respect to a ladder)**.**
Let $\mathcal{Y} = \prod_{j=1}^{s} \mathcal{Y}^j$ be a ladder on $[a, b]$. Then the *variation* of a function f with respect to $\mathcal{Y}$ is

$$V_{\mathcal{Y}}(f) = \sum_{y \in \mathcal{Y}} \left|\Delta(f; y, y_+)\right|,$$

where y_+ is the point obtained by taking the componentwise successor y_+^j in each dimension j.

The *Vitali variation* of a function is then the supremum of this ladder-based variation:

Definition 3.1.5 (Vitali Variation)**.**
The *Vitali variation* of a function f on the hyperrectangle $[a, b]$ is defined as

$$V_{[a,b]}(f) = \sup_{\mathcal{Y} \in \mathbb{Y}} V_{\mathcal{Y}}(f).$$

This variation captures the total fluctuation of f over axis-aligned rectangular subregions. It generalizes finite differences and serves as a foundation for defining the Hardy–Krause variation.

3.1.3 Definition of the Hardy–Krause Variation

The Hardy–Krause variation refines the Vitali variation by summing variations over all lower-dimensional axis-aligned faces of the unit hypercube. It is defined as:

Definition 3.1.6 (Hardy–Krause Variation)**.**
The *Hardy–Krause variation* of a function f on the hyperrectangle $[a, b]$ is defined as

$$V_{\mathrm{HK}}(f) = \sum_{v \subset \{1, \ldots, s\}} V_{[a^{-u}, b^{-u}]}\big(f(x^{-u}; b^u)\big).$$

This definition coincides with the Vitali variation but is typically used when $[a, b] = [0, 1]^s$.

Note that the Hardy–Krause variation reduces to the total variation in the univariate case. Importantly, $V_{\mathrm{HK}}(f) < \infty$ implies that f has bounded variation in all axis-aligned directions and projections.

Remark 3.1.7. The Hardy–Krause variation is a seminorm, vanishing for constant functions. It becomes a norm if the full-dimensional term is included explicitly.

3.1.4 Examples and Interpretation of Hardy–Krause Variation

While direct evaluation of the Hardy–Krause variation is intractable in general, upper bounds can be derived for smooth functions using mixed partial derivatives. For instance, for continuously differentiable functions, one can write:

$$V_{\mathrm{HK}}(f) \leq \int_{[0,1]^d} \left| \frac{\partial^d f}{\partial x_1 \cdots \partial x_d}(x) \right| dx + \sum_{i=1}^{d} V_{\mathrm{HK}}(f|_{x_i=1}),$$

where the right-hand side can be evaluated recursively and becomes exact if the mixed derivative is integrable and all lower-dimensional restrictions are sufficiently smooth.

Example 3.1.8. Let $f(x) = \prod_{i=1}^{d} x_i$, the d-dimensional monomial. Then f is smooth with bounded mixed derivatives and its Hardy–Krause variation is finite.

Example 3.1.9. Consider $f_{d,r}(x) = \left(\max\left\{\sum_{i=1}^{d} x_i - \frac{1}{2}, 0\right\}\right)^r$. For $d > r + 2$, this function has infinite Vitali and hence infinite Hardy–Krause variation. Such examples underline the limitations of QMC for functions with low smoothness or sharp nonlinearity [13, Prop. 16].

These examples illustrate the importance of regularity assumptions in applying the Koksma–Hlawka inequality. In practice, many physically motivated maps—such as solutions to elliptic or parabolic PDEs—do exhibit sufficient smoothness to ensure bounded Hardy–Krause variation, justifying the use of QMC error bounds.

3.2 The Koksma–Hlawka Inequality

The Koksma–Hlawka inequality is a central result in Quasi-Monte Carlo theory. It provides a deterministic error bound for numerical integration that combines two distinct concepts: the discrepancy of the point set and the variation of the integrand. This inequality formalizes the intuition that more uniformly distributed sample points lead to smaller integration errors—provided the integrand is regular enough.

3.2.1 Formal Theorem and Preconditions

The inequality relates the QMC integration error to the star discrepancy of the point set and the Hardy–Krause variation of the integrand.

Theorem 3.2.1 (Koksma–Hlawka Inequality). *Let $f\colon [0,1]^s \to \mathbb{R}$ be a function of bounded variation in the sense of Hardy–Krause and let $P = \{x_1, \ldots, x_N\} \subset [0,1]^s$ be a finite point set. Then the integration error satisfies*

$$\left| \frac{1}{N} \sum_{n=1}^{N} f(x_n) - \int_{[0,1]^s} f(x)\, dx \right| \leq D_N^*(P) \cdot V_{\mathrm{HK}}(f), \tag{3.1}$$

where $D_N^*(P)$ denotes the star discrepancy of the point set P and $V_{\mathrm{HK}}(f)$ is the Hardy–Krause variation of f.

Remark 3.2.2. This bound is deterministic and non-asymptotic. It applies to any dimension s and makes no probabilistic assumptions on the sampling procedure. However, it requires f to have bounded variation in the Hardy–Krause sense, which excludes some classes of discontinuous or highly irregular functions.

3.2.2 Role of Discrepancy and Function Variation

The two factors on the right-hand side of the inequality—discrepancy and variation—capture complementary aspects of integration error.

- The **star discrepancy** $D_N^*(P)$ measures how uniformly the point set P samples the unit cube. It is determined entirely by the geometry of the sample points and vanishes if the empirical distribution matches the uniform distribution.
- The **Hardy–Krause variation** $V_{\mathrm{HK}}(f)$ reflects the total directional variation of the integrand f along all coordinate axes and lower-dimensional projections. It penalizes irregularity and ensures that the integrand behaves well under uniform sampling.

Both quantities are required: even a low-discrepancy point set cannot guarantee small error if the function has unbounded variation. Conversely, even a smooth function may incur large error if the points cluster poorly.

Remark 3.2.3. The Koksma–Hlawka inequality can be seen as a product-type estimate: each component controls a different source of error and their product bounds the total integration error.

3.2.3 Interpretation: Bounding the Integration Error

The Koksma–Hlawka inequality provides valuable insight into the behavior of Quasi-Monte Carlo estimators:

- It motivates the use of low-discrepancy sequences such as Halton or Sobol', which minimize $D_N^*(P)$.
- It emphasizes the importance of function regularity—only functions with bounded Hardy–Krause variation are eligible for the bound.
- It establishes a worst-case, deterministic error guarantee, unlike the probabilistic bounds of Monte Carlo integration.

In practical terms, the inequality implies that for sufficiently smooth integrands and carefully constructed point sets, QMC methods can achieve integration errors of order

$$\mathcal{O}\left(\frac{(\log N)^s}{N}\right),$$

which significantly improves over the $\mathcal{O}(N^{-1/2})$ rate of standard Monte Carlo methods. This improvement, however, comes at the cost of stronger assumptions and increased sensitivity to dimensionality.

Remark 3.2.4. The deterministic nature of the bound makes it attractive in settings where reliability and reproducibility are crucial—such as scientific simulations or safety-critical computations.

Building upon this theoretical foundation, the next parts of this thesis turn from abstract principles to concrete applications. Part II investigates how QMC methods can enhance the training of neural networks, while Part III applies these techniques to the simulation of X-ray images in medical imaging.

Part II

Quasi-Monte Carlo Sampling for Neural Network Training

Chapter 4

Motivation and Problem Formulation

The second part of this thesis investigates the application of Quasi-Monte Carlos (QMCs) sampling techniques to enhance the training of Deep Neural Networks (DNNs) in *many-query problems* – computational tasks in which a parameterized map must be evaluated for a large number of inputs, each requiring costly simulations or numerical solutions of PDEs. This situation arises frequently in scientific computing, e.g. in uncertainty quantification, Bayesian inversion, optimal control and model calibration.

Such problems motivate the use of surrogate models $\hat{f}_\theta$, typically based on deep neural networks, which are trained offline on a set of inputs and then used for fast evaluation online. However, using random sampling for generating training data yields to star discrepancy $\mathcal{O}(N^{-1/2})$ as introduced in Part I.

To overcome this limitation, this thesis explores the use of low-discrepancy sequences as training data which fill the input domain more uniformly. Under mild regularity assumptions, they can significantly reduce the generalization error due to improved equidistribution properties.

4.1 Many-Query Problems in Scientific Computing

Many-query problems refer to scenarios in which a computationally expensive model $f(x)$ must be evaluated for many different parameter values $x \in \mathcal{X} \subset \mathbb{R}^s$. Typical applications include:

- **Uncertainty quantification (UQ)**: Compute statistics of $f(x)$ for uncertain x.
- **Optimal design/control**: Evaluate objective functions $f(x)$ for many designs x.
- **Inverse problems**: Repeatedly evaluate $f(x)$ during parameter inference.

The sheer cost of evaluating $f(x)$ (e.g., via numerical PDE solvers) motivates the use of surrogate models. Once trained, a surrogate $\hat{f}_\theta$ can provide rapid predictions for unseen inputs at a fraction of the computational cost.

4.2 The Role of Function Approximation

We formalize the problem as learning a map $f\colon [0,1]^s \to \mathbb{R}^m$ from data. The assumption of a unit hypercube domain reflects common practice in QMC theory and is justified via homeomorphic mappings from more general compact and simply connected input spaces.

The surrogate model $\hat{f}_\theta$ can be chosen from the class of deep neural networks and trained on data $\{(x_i, f(x_i))\}_{i=1}^N$. The aim is to approximate f with high accuracy while minimizing the number of required evaluations.

Deep Neural Networks are known to be universal approximators. Under mild conditions on the later introduced *activation functions* they proved to be able to approximate any continuous function on compact sets arbitrarily well [1].

4.3 Limitations of Random Sampling in Neural Network Training

Typically, training data is generated using Monte Carlo sampling, where points x_i are drawn independently and identically distributed (i.i.d.) according to a probability measure μ on $[0,1]^s$. Under this approach, statistical learning theory provides an estimate for the expected generalization error $\mathbb{E}[\mathcal{E}_G]$ as follows:

$$\mathbb{E}[\mathcal{E}_G] \leq \mathcal{E}_T + \mathcal{O}\left(\frac{1}{\sqrt{N}}\right),$$

where $\mathcal{E}_T$ is the empirical training error. However, as argued in [10], this rate is unsatisfactory in many-query problems because:

- A small generalization error requires a large N, which is costly due to expensive evaluations of $f(x)$.
- The constant in the bound depends on the variance and correlations in the training process, which are hard to control.

4.4 QMC-Based Surrogates: A Promising Alternative

Low-discrepancy sequences (Sobol', Halton, etc.) provide a way to sample the domain $[0,1]^s$ more uniformly than random sampling. In Quasi-Monte Carlo theory, the Koksma-Hlawka inequality (Theorem 3.2.1) gives an error bound of the form

$$|\mathcal{E}_G - \mathcal{E}_T| \leq V_{\mathrm{HK}}(|f - \hat{f}_\theta|) \cdot D_N^* \leq V_{\mathrm{HK}}(|f - \hat{f}_\theta|) \cdot C \cdot \frac{(\log N)^s}{N},$$

where V_{HK} denotes the Hardy–Krause variation and D_N^* is the star-discrepancy of the training points.

This suggests that QMC sampling can lead to significantly smaller generalization gaps – especially when f has bounded variation and the activation functions used in $\hat{f}_\theta$ are smooth.

Chapter 5

Theoretical Foundations of Deep Neural Networks

Deep Neural Networks (DNNs) have become a powerful class of surrogate models in scientific computing, particularly for high-dimensional and expensive-to-evaluate functions. This chapter introduces the mathematical structure and training procedure of DNNs, with a focus on activation functions, generalization theory and gradient-based optimization algorithms. It follows the framework established in [10].

5.1 Theoretical Foundations of Deep Neural Networks

The key feature enabling neural networks to approximate nonlinear functions is the application of scalar activation functions. We begin by introducing these components before defining the class of functions represented by Deep Neural Networks.

Definition 5.1.1 (Activation Function)**.**
An *activation function* is a measurable function $\sigma : \mathbb{R} \to \mathbb{R}$, typically nonlinear, that is applied componentwise to the output of each layer in a neural network. For a vector $z = (z_1, \dots, z_k)^\top \in \mathbb{R}^k$, the activation is defined as

$$\sigma(z) := (\sigma(z_1), \dots, \sigma(z_k))^\top.$$

Example 5.1.2 (Common Activation Functions).
Two widely used activation functions are illustrated in Figure 5.1. These are also the activation functions we employ in the experimental evaluations presented in Chapter 8.

- **ReLU (Rectified Linear Unit)**:

  $$\sigma(z) = \max\{0, z\}$$

  ReLU is piecewise linear and unbounded. It is computationally efficient and commonly used in deep architectures.

- **Sigmoid**:

  $$\sigma(z) = \frac{1}{1 + e^{-z}}$$

The sigmoid function maps real numbers to $(0, 1)$ and is differentiable everywhere. It is often used in theoretical analysis and binary classification tasks.

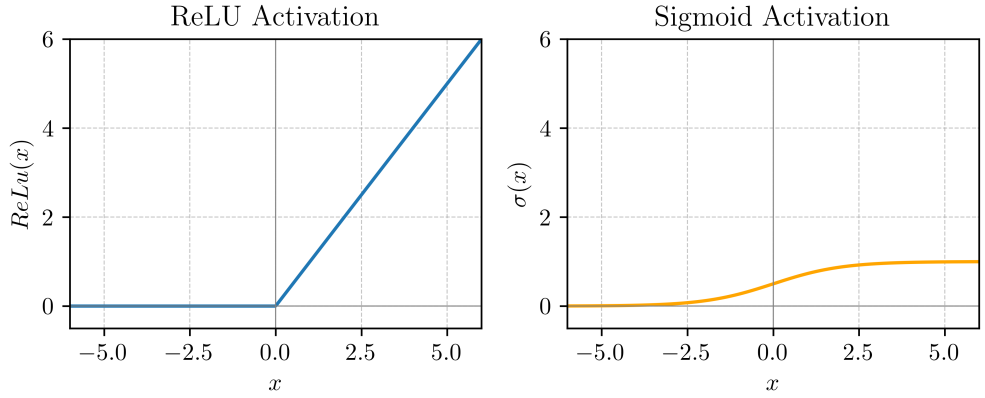


FIGURE 5.1: Comparison of ReLU and Sigmoid activation functions.

With an activation function in place, we now define the class of functions that a fully connected feedforward neural network can represent.

Definition 5.1.3 (Class of Deep Neural Network Functions)**.**
Let $L \in \mathbb{N}$ and $\mathbf{d} = (d_0, d_1, \ldots, d_L)$ be a sequence of positive integers, where $d_0 = s$ is the input dimension and $d_L = 1$ the output dimension. A Deep Neural Network (DNN) of *depth* L and *layer widths* $d_1, \ldots, d_{L-1}$ is a function $f_\theta : \mathbb{R}^s \to \mathbb{R}$ of the form

$$f_\theta(y) = A_L \circ \sigma \circ A_{L-1} \circ \cdots \circ \sigma \circ A_1(y),$$

where each *layer map* $A_l : \mathbb{R}^{d_{l-1}} \to \mathbb{R}^{d_l}$ is an affine transformation

$$A_l(z) = W_l z + b_l,$$

with weights $W_l \in \mathbb{R}^{d_l \times d_{l-1}}$ and biases $b_l \in \mathbb{R}^{d_l}$. The activation function $\sigma : \mathbb{R} \to \mathbb{R}$ is applied component-wise.
The collection of all trainable parameters is denoted by

$$\theta = \{(W_l, b_l)\}_{l=1}^{L},$$

and fully determines the function f_θ. The set of all such realizable functions for a given architecture and activation σ is denoted by $\mathcal{F}_{L,\mathbf{d},\sigma}$.

The architectural parameters are interpreted as follows:

- The *depth* L is the number of affine layers.
- The *width* d_l specifies the number of neurons in layer l.
- The *layer structure* is encoded in the tuple $\mathbf{d}$.

In the course of this work, we assume that all hidden layers have the same number of neurons, i.e., the network has constant width m for all layers $l = 1, \ldots, L-1$. In Figure 5.2, a fully connected DNN with constant width is illustrated.

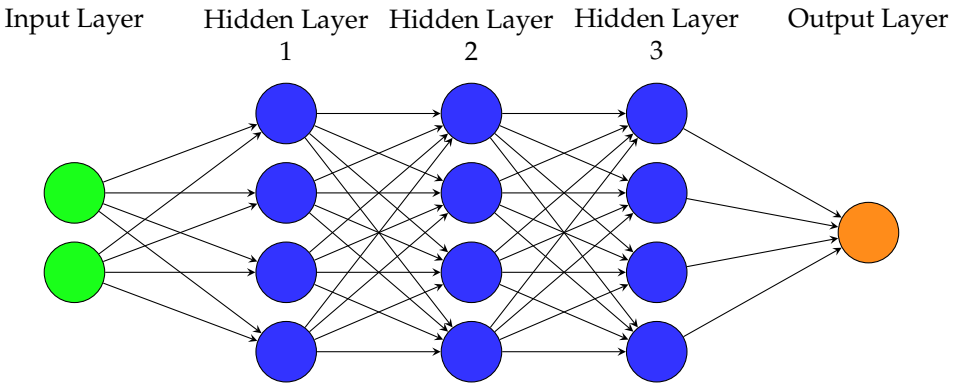


FIGURE 5.2: Schematic of a fully connected Deep Neural Networks with constant width $m = 4$ and depth $L = 3$ layers.

Remark 5.1.4. For the layer structure $d = (d_0, ..., d_L)$, the total number of trainable parameters (weights and biases) of *theta* in the network is given by

$$N_\theta = \sum_{l=1}^{L} (d_{l-1} \cdot d_l + d_l) = (L-2) \cdot m^2 + (L+s) \cdot m + 1.$$

5.2 Training and Generalization Error

Supervised training of a DNN requires minimizing the discrepancy between the network output $f_\theta(y)$ and a ground truth map $L(y)$ on a given training set $\{y_i\}_{i=1}^N$.

Definition 5.2.1 (Training Error).
The empirical risk or training error is defined by

$$\mathcal{E}_T(\theta) = \frac{1}{N} \sum_{i=1}^{N} |L(y_i) - f_\theta(y_i)|.$$

Definition 5.2.2 (Generalization Error).
The generalization error (or population risk) is the expected error under the data distribution μ:

$$\mathcal{E}_G(\theta) = \int_{\mathcal{Y}} |L(y) - f_\theta(y)| \, \mathrm{d}\mu(y).$$

5.3 Optimization and Training Procedures

Training a DNN corresponds to solving the following regularized minimization problem:

$$\theta^* = \arg\min_{\theta \in \Theta} \left[\mathcal{E}_T(\theta) + \lambda \cdot R(\theta) \right], \tag{5.1}$$

where $R(\theta)$ is a regularization term (e.g., ℓ^2 weight penalty) and $\lambda > 0$ controls its strength.

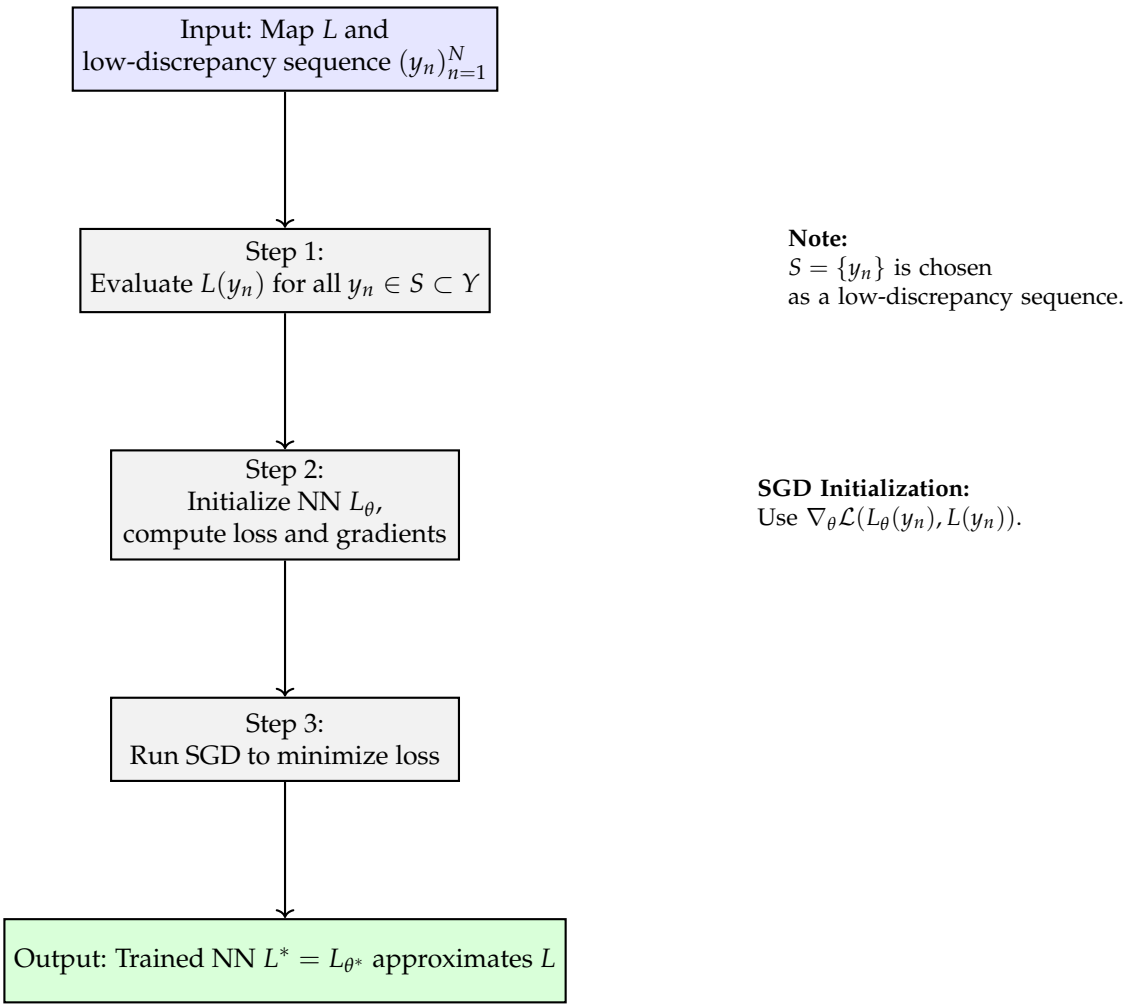


FIGURE 5.3: Workflow of Algorithm 2.2: Deep learning of parameters-to-observable map using low-discrepancy sampling and SGD-based training.

5.3.1 Stochastic Gradient Descent (SGD)

In practice, (5.1) is minimized via stochastic gradient descent:

$$\theta_{t+1} = \theta_t - \eta \cdot \nabla_\theta \mathcal{E}_{\mathrm{T}}^{(B)}(\theta_t),$$

where η is the learning rate and $\mathcal{E}_{\mathrm{T}}^{(B)}$ is the mini-batch loss.

5.3.2 Adam Optimizer

The Adam algorithm improves on SGD by incorporating adaptive learning rates and momentum:

Algorithm 1 Adam Optimizer (simplified)

Initialize θ_0, $m_0 = 0$, $v_0 = 0$
for $t = 1$ to T **do**
  $g_t \leftarrow \nabla_\theta \mathcal{E}_T(\theta_{t-1})$
  $m_t \leftarrow \beta_1 m_{t-1} + (1-\beta_1) g_t$
  $v_t \leftarrow \beta_2 v_{t-1} + (1-\beta_2) g_t^2$
  $\hat{m}_t \leftarrow m_t / (1-\beta_1^t)$
  $\hat{v}_t \leftarrow v_t / (1-\beta_2^t)$
  $\theta_t \leftarrow \theta_{t-1} - \eta \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$
end for

Remark 5.3.1. The Adam optimizer is particularly effective in early training and in high-dimensional, noisy settings. Its convergence properties, however, require careful tuning of β_1, β_2 and ϵ.

Chapter 6

Theoretical Foundations of QMC-Based Learning

- Adapting QMC theory to supervised learning
- Hardy-Krause variation of the loss function
- Generalization error bounds using low-discrepancy samplin
-
- Comparison to MC-based bounds and practical implication
-
- Validity conditions and smoothness assumptions

$$
\begin{aligned}
V_{\mathrm{HK}}(f) = V_{\mathrm{HK}}(f;a,b) &= \sum_{u\subset\{1,\dots,s\}} V_{[a^{-u},b^{-u}]} f(x^{-u}:b^{u}) \\
&= \sum_{u\subset\{1,\dots,s\}} \sup_{\mathcal{Y}\in\mathbb{Y}} V_{\mathcal{Y}}\big(f(x^{-u}:b^{u})\big) \\
&= \sum_{u\subset\{1,\dots,s\}} \sup_{\mathcal{Y}\in\mathbb{Y}} \sum_{y\in\mathcal{Y}} \big|\Delta\big(f(x^{-u}:b^{u});y,y_{+}\big)\big| \\
&= \sum_{u\subset\{1,\dots,s\}} \sup_{\mathcal{Y}\in\mathbb{Y}} \sum_{y\in\mathcal{Y}} \Big|\sum_{v\subseteq\{1,\dots,s\}} (-1)^{|v|} f((y^{v}:y_{+}^{-v})^{-u}:b^{u})\Big|
\end{aligned}
$$

For the definitions of Variation we follow [13] and start by introducing the mixed power notation:

Definition 6.0.1 (Mixed Component Selection)**.**
Let $\boldsymbol{a},\boldsymbol{b}\in\mathbb{R}^s$ and let $u\subseteq\{1,\dots,s\}$ be an index set. We define

$$
\boldsymbol{c} := \boldsymbol{a}^{u}:\boldsymbol{b}^{-u} \quad \text{with} \quad c_j := \begin{cases} a_j, & \text{if } j\in u, \\ b_j, & \text{if } j\notin u. \end{cases}
$$

Hereby the term $-u$ denotes the complement of u in $\{1,\dots,s\}$. Further the alternating sum is defined.

Definition 6.0.2 (Alternating sum)**.**
The *alternating sum* on s dimensions with respect to a function on the hyperrectangle $[a, b]$ is defined as

$$\Delta(f; a, b) = \sum_{v \subseteq \{1,\dots,s\}} (-1)^{|v|} f(a^v : b^{-v}).$$

An essential concept in this context is the notion of a *ladder* which we define on a certain hypercube $[a, b]$.

Definition 6.0.3 (Ladder)**.**
A *ladder* on the hypercube $[a, b]$ is a set of points $\mathcal{Y} = \{y_1, \dots, y_k\} \subset [a, b]$ such that for each $i < j$, the point y_i is less than or equal to y_j in every coordinate. Formally, this means that for all $i < j$ and for all $l \in \{1, \dots, s\}$, we have $y_i^l \leq y_j^l$.

The space of all such ladders on the hypercube is denoted by $\mathbb{Y}$. Next, the *variation with respect to a ladder* is defined.

Definition 6.0.4 (Variation with respect to a ladder)**.**
The *variation* of a function f on the hyperrectangle $[a, b]$ *with respect to a ladder* $\mathcal{Y} = \prod_{j=1}^{s} \mathcal{Y}^j$ is defined as

$$V_{\mathcal{Y}}(f) = \sum_{y \in \mathcal{Y}} \left|\Delta(f; y, y_+)\right|,$$

where y_+ is the point obtained by incrementing each coordinate of y with y_+^j to be the successor of y_j in $\mathcal{Y}^j$.

Accordingly the Variation by Vitali is defined as the supremum of the variation over all ladders $\mathcal{Y} \in \mathbb{Y}$.

Definition 6.0.5 (Vitali Variation)**.**
The *Vitali variation* of a function f on the hyperrectangle $[a, b]$ is defined as

$$V(f) = V(f; a, b) = \sup_{\mathcal{Y} \in \mathbb{Y}} V_{\mathcal{Y}}(f).$$

Here $\mathbb{Y}$ denotes the set of all ladders on the hyperrectangle $[a, b]$.

The Hardy–Krause variation is a specific case of the Vitali variation, where the supremum is taken over all ladders in the unit hypercube $[0, 1]^s$.

Definition 6.0.6 (Hardy–Krause Variation)**.**
The *Hardy–Krause variation* of a function f on the hyperrectangle $[a, b]$ is defined as

$$V_{\mathrm{HK}}(f) = V_{\mathrm{HK}}(f; 0, 1) = \sup_{\mathcal{Y} \in \mathbb{Y}} V_{\mathcal{Y}}(f).$$

Chapter 7

QMC-Based Deep Learning Algorithm

- Algorithm description (based on Mishra & Rusch Algirthm 2.2)
- Network Architecture, activation functions and regularization
- QMC vs. MC sampling: sample complexity and efficiency
- Discussion of smooth vs. non-smooth activations (e.g. ReLU vs. tanh)
- Practical considerations: dimensionality, optimal choices

Chapter 8

Empirical Enaluation and Discussion

- Overview of benchmark tasks:
  - Synthetic function approximation
  - ODE simulation (projectile motion)
  - Computational finance (option pricing)
  - CFD (airfoil flow)
- Comparison of DLsob (QMC) and DLrand (MC) Performance
- Error metrics and convergence plots
- Limitations, edge cases and future directions

Part III

X-ray Simulation using QMC Methods

Chapter 9

Motivation and Problem Statement

9.1 Computed Tomography Imaging

CT imaging is a powerful medical imaging technique that provides detailed cross-sectional images of the body by combining multiple X-ray projections taken from different angles. This technique is widely used for diagnostic purposes, allowing clinicians to visualize internal structures with high spatial resolution.

The process involves rotating an X-ray source around the patient while simultaneously capturing X-ray projections on a detector array. Each projection represents the cumulative attenuation of X-rays as they pass through various tissues, influenced by the density and composition of the materials they encounter.

The collected data is then reconstructed into a three-dimensional volume using advanced algorithms, such as filtered backprojection or iterative reconstruction methods. These algorithms convert the raw projection data into cross-sectional images, which can be further processed to enhance contrast, reduce noise, and improve overall image quality.

To achieve accurate reconstructions, it is crucial to acquire high-quality X-ray projection images. However, one of the most significant challenges in CT imaging is the presence of scattered radiation - such as Comton and Rayleigh scattering - which can lead to artifacts and distortions in the reconstructed images.

Throughout this thesis, the term *scatter reduction in CT* refers to the mitigation of scatter-induced artifacts in the two-dimensional X-ray projection images acquired by the detector array during a CT scan. These projections serve as the input for reconstructing the final three-dimensional volume. Given the complexity of the full reconstruction process, the focus of this work is limited to analyzing and correcting scatter effects at the level of the two-dimensional projection data.

9.2 X-ray Imaging and the Challenge of Scatter

X-ray CT relies on measuring the attenuation of X-rays along straight-line paths through a phantom. Here, the X-ray beam is further abstracted as individual photons with defined initial energies and directions. The photon trajectories typically extend from an X-ray source $\mathcal{S}$ to a detector element $\mathcal{D}$, forming the path:

$$l = \overrightarrow{\mathcal{SD}}$$

Under idealized conditions, the attenuation process is governed by the *Lambert-Beer law*, which describes an exponential decay in intensity I as the beam interacts with the material. As stated in [9, Chap. 7], this relationship is given by:

$$I = I_0(E) \cdot \int_0^{E_{\max}} \exp\left(- \int_{\mathcal{S}}^{\mathcal{D}} \mu(x, E)\, dx \right) dE \tag{9.1}$$

Here, $I_0(E)$ denotes the incident intensity of X-rays with initial energy E at the source $\mathcal{S}$, and $\mu(x, E)$ represents the linear attenuation coefficient at spatial location x along the path l, for energy E. The resulting intensity I is measured at the detector element $\mathcal{D}$.

Although exponential attenuation along straight-line paths such as $\overrightarrow{\mathcal{SD}}$ is the idealized model for X-ray imaging, this assumption is systematically violated in practice. As X-rays traverse the scanned phantom, many photons undergo scattering interactions - such as Compton or Rayleigh scattering - that alter their trajectories. Despite deviating from the primary path, these scattered photons may still reach the detector, adding unintended signal components. As a result, the measured intensities no longer represent pure line integrals of the attenuation map. This discrepancy introduces nonlinear errors and visible artifacts in the reconstructed image, ultimately degrading both visual quality and quantitative accuracy.

Scattered radiation is a major source of image artifacts - such as cupping and streaks - and reduces both spatial and contrast resolution. These effects not only degrade visual image quality but also compromise the accuracy of quantitative measurements, such as Hounsfield units μ_*, which are used for clinical interpretation, such as tissue characterization [9, Chap. 8]. Hounsfield units represent a normalized attenuation relative to the attenuation of water:

$$\mu_* = \left(\frac{\mu}{\mu_{\text{water}}} - 1 \right) \cdot 1000 \tag{9.2}$$

The impact of scatter becomes especially pronounced in modern CT systems using high-energy X-rays or large-area flat-panel detectors, where scatter may dominate the measured signal. As such, accurate modeling and correction of scatter are essential for achieving high-fidelity CT images, particularly in clinical applications where precision and reliability are paramount [9].

9.3 Scatter Correction Methods for X-ray Imaging

9.3.1 Overview of Scatter Correction Techniques

In X-ray imaging, scattered photons are a major source of image artifacts and quantitative inaccuracies. To mitigate these effects, a range of computational scatter correction methods has been developed. These approaches can be broadly classified into the following categories:

- **Empirical and Analytical Methods:**
  These include techniques such as primary modulation, convolution-based correction, and energy windowing. Scatter is typically estimated using simplified

models or empirical kernels, often assuming a smooth background distribution, and then subtracted from the measured signal. While computationally efficient, these methods rely on assumptions regarding the spatial and energy distribution of scattered photons. As a result, they may fail to accurately model complex scatter phenomena in heterogeneous anatomical structures.

- **Physics-Based Models:**
  These methods aim to provide a more accurate representation of the underlying photon transport physics, including scattering phenomena. Among them, MC simulation is regarded as the most rigorous and comprehensive technique due to its ability to statistically model complex photon interactions without relying on simplifying assumptions. In such simulations, primary and scattered photon contributions are detected separately, allowing the estimated scatter signal to be subtracted from measured data in order to restore image fidelity.

In 2011, Sisniega et al. [19] already demonstrated promising results using computationally expensive MC simulations of CT scans of a cylindrical phantom with two bone inserts. The results showed significant improvements in image quality, as illustrated by Sisniega et al. in Figure 9.1.

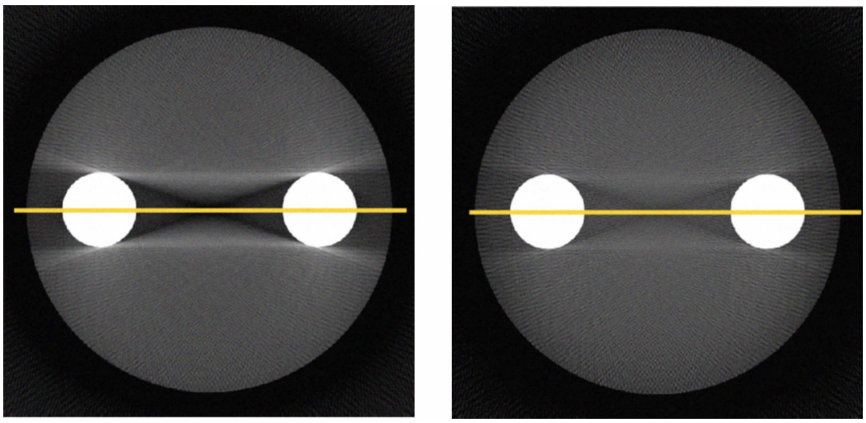


FIGURE 9.1: Cone-Beam Computed Tomography (CBCT) of a 70 mm soft-tissue cylinder with two bone inserts. Right: Simulation of a X-ray image with scattering, exibiting typical cupping and streak artifacts. Right: Image after scatter correction applying MC techniques for the simulation of 5×10^6 photons, showing improved uniformity and quantitative accuracy. Right: Image without correction, exhibiting typical cupping and streak artifacts. (Figure adapted from [19], ©2011 IEEE)

9.3.2 Monte Carlo Simulation

To achieve scatter correction using MC simulation, a three-dimensional phantom is first estimated by analyzing the X-ray projections. Then, an MC simulation is performed to estimate the scatter signal $I_{\text{scattered}}$ for each detector pixel. This estimated scatter signal is then subtracted from the measured intensity I_{measured}:

$$I_{\text{corrected}} = I_{\text{measured}} - I_{\text{scattered}} \tag{9.3}$$

The MC simulation is a powerful computational technique that follows physics-based principles to model the transport of photons through matter. It is widely recognized as the gold standard for scatter correction in CT and related imaging modalities. This status is attributed to several key factors:

- **Physical Accuracy:**
  MC methods simulate the stochastic nature of photon interactions - including Compton and Rayleigh scattering, photoelectric absorption, and multiple scattering events - based on fundamental physical cross-sections and material properties.
- **Comprehensive Modeling:**
  Unlike analytical or empirical methods, MC simulations can account for complex geometries, heterogeneous materials and realistic X-ray spectra, providing highly accurate estimates of the scatter signal.
- **Validation Benchmark:**
  Due to their accuracy, MC-based scatter estimates are routinely used as reference standards for validating and benchmarking faster, approximate correction methods.

9.4 High-Level Overview of the Monte Carlo Simulation

The MC simulation of photon transport for scatter correction typically involves the following key steps [19]:

1. **Photon Emission:**
   Photons are emitted from a virtual X-ray source, with their energies sampled from a predefined source spectrum.
2. **Photon Tracking:**
   Each photon is tracked as it propagates through the object. At every step, the probability of interaction (either scattering or absorption) is determined by the local material properties and corresponding interaction cross-sections.
3. **Interaction Sampling:**
   Upon interaction, the event type (e.g., Compton scattering, Rayleigh scattering, or photoelectric absorption) and the resulting change in photon direction and energy are sampled from the relevant probability distributions.
4. **Detection:**
   Photons that reach the detector—either unscattered or after one or more scattering events—are recorded. The simulation distinguishes between primary and scattered photons, thereby enabling an accurate estimation of the scatter contribution at each detector element.
5. **Statistical Averaging:**
   By simulating a large number of photons, the method builds a statistically robust estimate of the scatter distribution. The accuracy of the result increases with the number of simulated photons and ultimately converges toward a stable intensity distribution.

The schematic flow of the Monte Carlo simulation process is summarized in Figure 9.2.

Monte Carlo Photon Transport Algorithm

- **Initialization:** Define geometry, material composition, and source spectrum.
- **Loop over photons:**
  - Sample the step size to the next interaction point.
  - Move the photon; check for boundary crossings.
  - Sample the interaction type; update direction and energy.
  - If the photon reaches the detector, record the event.
  - If the photon is absorbed or exits the system, terminate its trajectory.
- **Aggregation:** Compute both scatter and primary intensity distributions.

FIGURE 9.2: Pseudocode representation of the Monte Carlo algorithm for photon transport in scatter modeling.

9.5 Monte Carlo Methods: Benefits & Drawbacks

MC simulations are widely regarded as the gold standard for scatter correction in computed tomography due to their ability to model photon-matter interactions from first principles. These simulations accurately reproduce all relevant physical scattering phenomena - including Compton and Rayleigh scattering - and can accommodate arbitrarily complex geometries and heterogeneous material compositions. Owing to this high degree of physical fidelity, MC-based methods yield highly accurate scatter estimates and are commonly employed as reference standards for evaluating and validating alternative correction approaches.

However, the high accuracy of Monte Carlo simulations comes at the cost of substantial computational effort. Producing low-noise scatter estimates requires the simulation of a large number of photon histories to ensure statistical convergence. Consequently, the runtime scales with the desired level of accuracy and may range from several minutes to multiple hours or even days, depending on the complexity of the scene and the available computational resources. This computational burden poses a major limitation, particularly in applications involving large datasets or iterative reconstruction workflows.

To address the computational inefficiency of slow-converging MC methods, recent research has explored strategies to accelerate physically accurate photon transport simulations. Among these, the application of QMC methods has gained increasing attention. By replacing random sampling with deterministic low-discrepancy sequences, QMC techniques can achieve significantly faster convergence while maintaining comparable accuracy. Recent QMC-based scatter correction algorithms have demonstrated runtime reductions of multiple orders of magnitude, thereby enabling high-fidelity simulations even in time-sensitive or resource-constrained environments [7, 8, 2].

Chapter 10

Physical laws of Photon Simulation

This chapter introduces the physical and mathematical principles underlying the simulation of X-ray imaging. Central to this is the modeling of individual photon interactions with matter, including attenuation and scattering processes. These interactions are inherently stochastic due to the quantum nature of radiation-matter interactions. The stochastic nature of photons and their interactions with matter is modelled using monte Carlo methods of uniformly distributed values between 0 and 1, which are then transformed to follow the physical probability distributions relevant to the specific interactions being simulated. This approach allows for a realistic representation of photon transport and interaction within the imaging system.

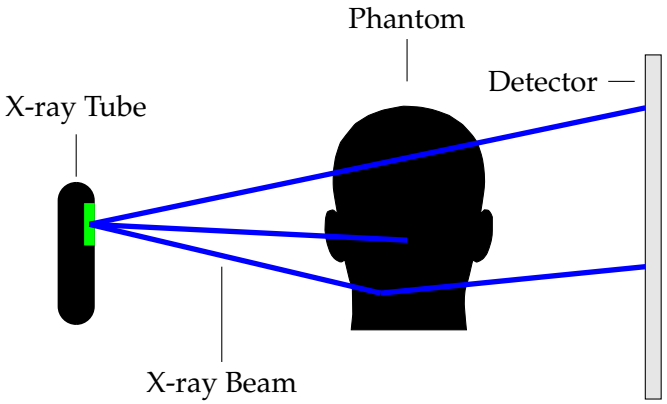


FIGURE 10.1: Schematic illustration of photon transport in X-ray imaging.

In a typical simulation workflow, individual photons are emitted from the X-ray tube and propagate through air before reaching the phantom. Upon entering the phantom, they may interact with the material through scattering or absorption, depending on the local composition of the tissue and photon energy. Only a fraction of the photons will exit the phantom, continue their path through air, and ultimately reach the detector, contributing to the final image formation.

The simulation framework described here forms the basis for all subsequent analyses presented in this thesis.

10.1 X-Ray Tube

X-ray beams are generated within evacuated glass tubes containing several critical components that convert electrical energy into X-ray photons and heat. The X-ray tube acts as an energy converter, where the vast majority of the input energy is transformed into heat and only a small fraction becomes useful radiation. The following Figure 10.2 illustrates the basic construction of an X-ray tube as in [17]

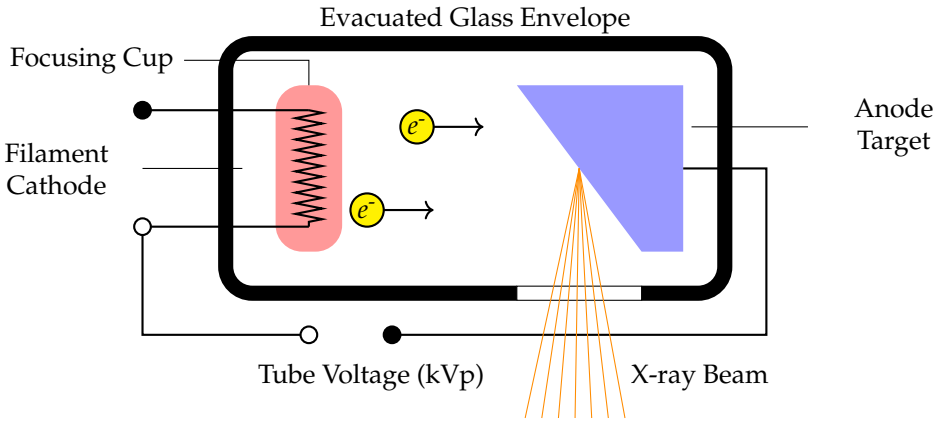


FIGURE 10.2: Schematic of an X-ray tube.

10.1.1 Photon Generation

To capture this randomness, Monte Carlo methods are employed. In such simulations, random numbers—typically sampled uniformly from the interval $[0, 1]$ are transformed into physically meaningful quantities according to the relevant probability distributions. This probabilistic modeling enables realistic simulation of photon transport and forms the basis for the analyses presented in this thesis.

The key components are:

- **Filament (Cathode):** A tungsten wire filament serves as the electron source through thermionic emission. When a current of approximately 3-6 A passes through it, the filament reaches incandescence, releasing electrons from its surface. These free electrons form a cloud near the cathode until they are accelerated toward the anode by the applied high voltage.
- **Focusing Cup:** The filament is embedded in a negatively charged, nickel-made focusing cup. Its function is to electrostatically shape and direct the electron stream toward the anode's focal spot, thereby influencing the resolution and size of the resulting X-ray beam.
- **Anode Target:** The anode consists of a tungsten target, often embedded in a copper support. Tungsten is used for its high atomic number and melting point, enhancing X-ray production and durability. The copper base improves heat dissipation. Typically, less than 1% of the electron energy is converted into X-rays, with the remainder generating heat that must be effectively managed.
- **Evacuated Glass Envelope:** All components are sealed within a borosilicate glass or metal-ceramic housing evacuated to a low pressure (typically 10^{-5} to 10^{-7} hPa). The vacuum allows unimpeded electron flow and prevents arcing.

The housing is usually immersed in insulating oil to provide thermal and electrical isolation.

When high-energy electrons strike the tungsten target, X-ray photons are produced through two primary mechanisms:

- **Bremsstrahlung (Braking Radiation):** This accounts for approximately 80% of X-ray production. When electrons pass close to tungsten nuclei, they are decelerated by the electrostatic attraction, causing them to lose kinetic energy that is emitted as X-ray photons. This process produces a continuous spectrum of X-ray energies from near zero up to the maximum electron energy.
- **Characteristic Radiation:** This occurs when high-energy electrons knock inner shell electrons from tungsten atoms. When outer shell electrons drop down to fill these vacancies, they emit X-rays with discrete, characteristic energies specific to tungsten. Therefore peaks are occuring at the difference of the binding energies of the electron shells. For reference, the atomic model of tungsten is given in Figure 10.7 and the approximate binding energies of the electron shells in Table 10.1.

Shell / Subshell	Binding Energy (keV)
K	69.5
L_1	12.1
L_2	11.5
L_3	10.2
M_1	2.82
M_2	2.30
M_3	2.15
N_1	0.43
N_2	0.32
N_3	0.22

TABLE 10.1: Approximate Electron Binding Energies of Tungsten (W, Z = 74)

In Table 10.1 only binding energies of the K, L and M shells are given, since these are the most relevant for X-ray production. The binding energies of the N shell and higher shells are negligible in this context due to their low binding energies.

Although it is not necessary to understand every technical detail of the X-ray apparatus for the purposes of simulation, I found it important to briefly present the fundamental working principles of the X-ray tube. This background allows one to appreciate how key simulation parameters for photon generation - specifically the tube voltage and the cathode material - influence the resulting X-ray spectrum and photon behavior.

Figure 10.4 below shows an resulting energy spectrum for an X-ray tube with a tungsten cathode operated at 100 kVp. It illustrates the resulting distribution of photon energies, which is shaped by both the material and the applied voltage. The intensity of the different photon energies is measured in *spectral fluence* in $\mathrm{cm}^{-2}\,\mathrm{keV}^{-1}$,

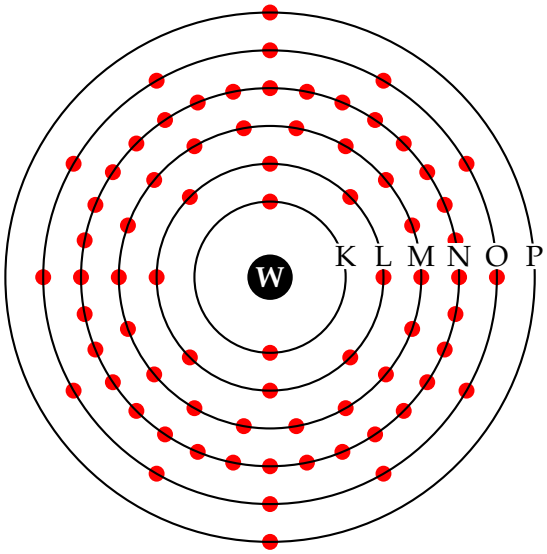


FIGURE 10.3: Bohr model of the tungsten atom (W, Z = 74) with electron shells and subshells.

which describes the number of X-ray photons per unit area per unit energy interval. It certainly shows the two components of the X-ray spectrum: the continuous bremsstrahlung spectrum and the characteristic radiation peaks:

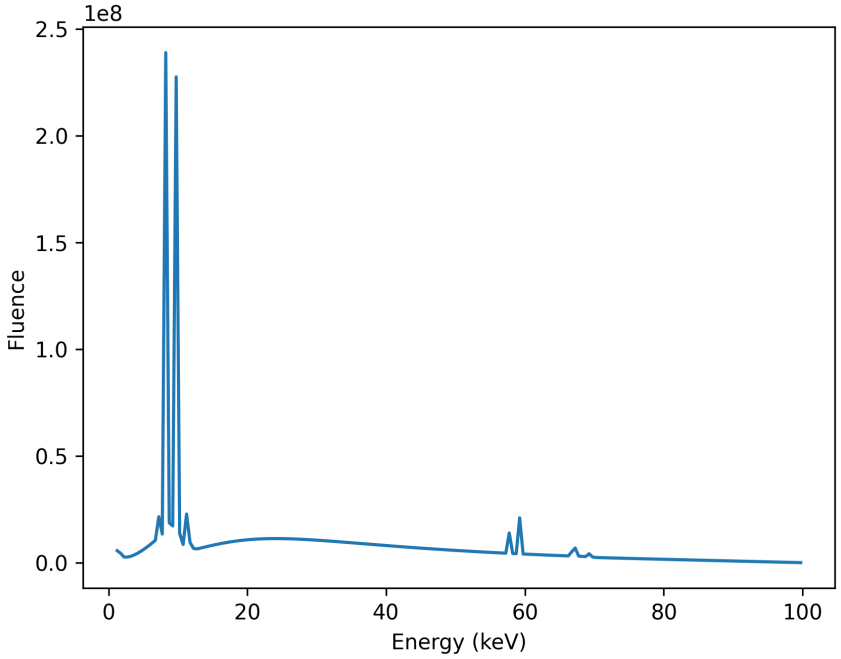


FIGURE 10.4: X-ray spectrum for a tungsten cathode at 100 kVp showing the continuous bremsstrahlung spectrum and characteristic peaks. Build with *SpekPy* [16]

10.1.2 Filter

Low energy X-ray photons contribute little to image formation but significantly increase patient dose. Therefore, X-ray tubes are often equipped with filters to selectively attenuate these low-energy photons while allowing higher-energy photons to pass through. This process, known as beam hardening, improves image quality by reducing scatter and enhancing contrast. Common filter materials include aluminum, copper and molybdenum, which are chosen based on their atomic number and thickness to effectively absorb low-energy photons while minimizing the impact on higher-energy photons [17].

As can be seen in Figure 10.5, the filter is placed at the X-ray tube margin such that the photons hit the filter after the anode target and before leaving the X-ray tube. Followingly X-rays pass through the filter before reaching the patient.

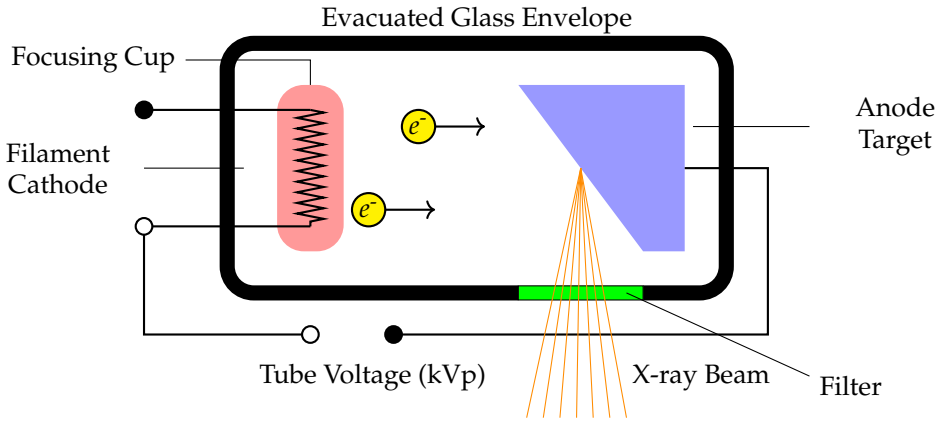


FIGURE 10.5: Schematic of an X-ray tube with Filter.

Throughout this thesis, a filter consisting of 0.4 mm Tin (Sn) and 0.1 mm Copper (Cu) is used for the simulations. This filter is chosen to represent a realistic X-ray tube filter that is commonly used in clinical practice [20]. The resulting spectrum is shown in Figure 10.6.

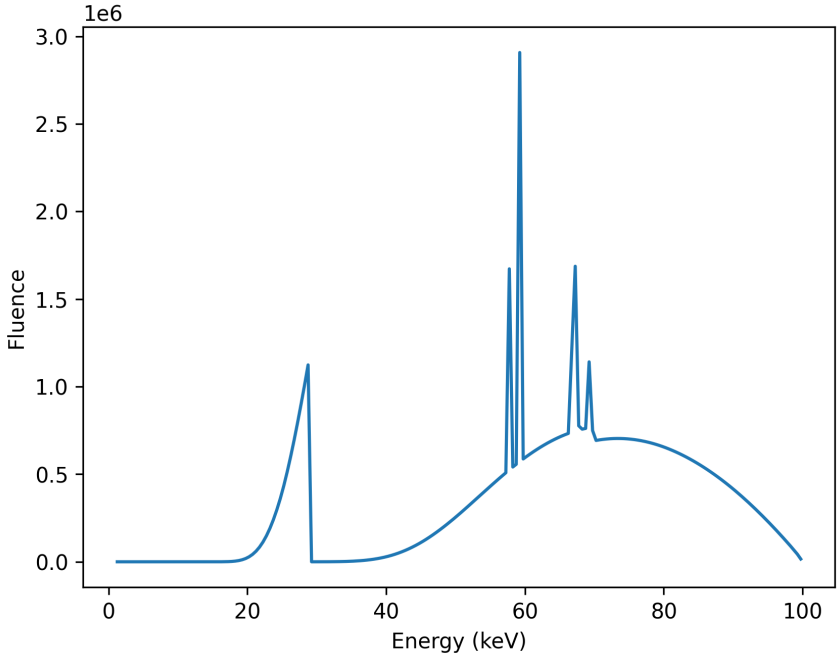


FIGURE 10.6: X-ray spectrum for a tungsten cathode at 100 kVp with a 0.4 mm showing the continuous bremsstrahlung spectrum and characteristic peaks. Build with *SpekPy* [16]

10.2 Photon Generation

In the context of Monte Carlo simulations, photons are generated from the X-ray tube source, which is typically modeled as a point source emitting photons within a conical beam. The emission cone is defined by a half-angle Θ, which determines the angular distribution of emitted photons. – The emitted photons are characterized by two key properties: their direction and energy.

10.2.1 Photon Energy

The photon energy is sampled from the X-ray tube spectrum as described in Section 10.1.2. In the simulations referenced in this thesis the spectra are generated utilizing *SpekPy* [16, 18]. The energies of the resulting photons are sampled via inverse transform sampling from the normalized spectrum.

Inverse transform sampling [11] is a method to generate random samples from a target distribution using uniformly distributed random variables, such as those generated by a Quasi-Monte Carlo sequence.

Definition 10.2.1 (Inverse Transform Sampling)**.**
Let $F_X : \mathbb{R} \to [0,1]$ be the cumulative distribution function (CDF) of a continuous, strictly increasing random variable X. The method of *inverse transform sampling* generates a realization of X by the following procedure:

1. Generate a sample U from the uniform distribution on the unit interval, i.e., $U \sim \mathcal{U}(0,1)$.
2. Compute the value $X := F_X^{-1}(U)$, where F_X^{-1} denotes the inverse of the CDF F_X.

Theorem 10.2.2.
Let F_X be a continuous and strictly increasing cumulative distribution function and let $U \sim \mathcal{U}(0,1)$. Then the random variable

$$X := F_X^{-1}(U)$$

has cumulative distribution function F_X, i.e.,

$$\mathbb{P}(X \leq x) = F_X(x), \quad \textit{for all } x \in \mathbb{R}.$$

Proof.
Since F_X is continuous and strictly increasing, its inverse F_X^{-1} exists. For any $x \in \mathbb{R}$, we compute:

$$\mathbb{P}(X \leq x) = \mathbb{P}(F_X^{-1}(U) \leq x) = \mathbb{P}(U \leq F_X(x)) \overset{\text{since } F_X \text{ is strictly decreasing}}{=} F_X(x),$$

because $U \sim \mathcal{U}(0,1)$ and thus $\mathbb{P}(U \leq u) = u$ for all $u \in [0,1]$. This shows that X has CDF F_X. □

10.2.2 Photon Direction

The direction of each emitted photon is sampled uniformly within a conical emission cone defined by the half-angle Θ. For the simulation a spherical alignment of the X-ray tube is assumed, such that the beam is oriented along the vector $\vec{v} = (v_1, v_2, v_3)$ in the Cartesian coordinate system.

Followingly, the direction of the emitted photon is sampled based on two random values $u_1, u_2 \in [0,1)$ as follows:

1. Sample a random angle θ uniformly from the interval $[0, \Theta]$ with u_1:

$$\theta = u_1 \cdot \Theta$$

2. Sample a random azimuthal angle ϕ uniformly from the interval $[0, 2\pi)$ with u_2:

$$\phi = u_2 \cdot 2\pi$$

3. Compute the direction vector $\vec{d}$ of the photon as:

$$\vec{d} = (\sin(\theta)\cos(\phi), \sin(\theta)\sin(\phi), \cos(\theta))$$

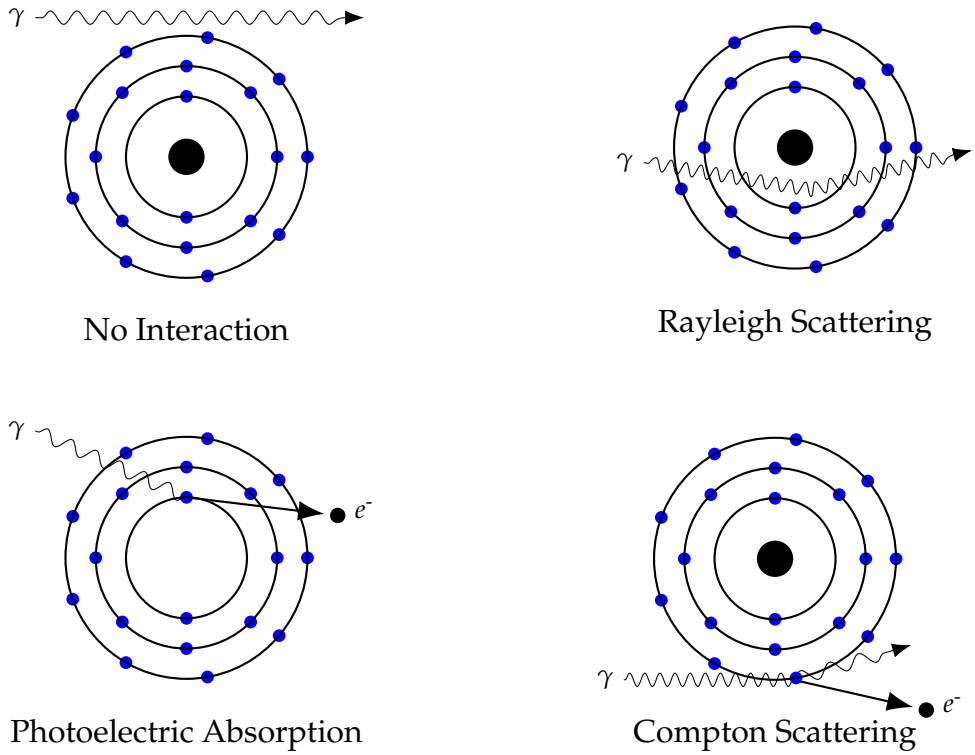


FIGURE 10.7: Principles from photon interaction with matter similar to [9, Chap. 7]

10.3 Scattering and Attenuation

This section provides the basic concepts of photon interaction with matter. The interaction relevant for medical imaging results in a reduction of radiation intensity, which corresponds to a decreased number of photons reaching the detector. Hereby X-ray photons may be fully absorbed by *photoelectric absorption* or undergo either *elastic scattering* (Rayleigh) or *inelastic scattering* (Compton) as they interact with matter.

The attenuation of X-ray photons arises from physical processes that alter their number, direction, or energy as they interact with matter. These interactions occur at the level of individual photons and are highly dependent on the photon energy. This section presents an overview of the primary interaction mechanisms relevant to attenuation like in [9].

For correctness, in the simmulations it is further assumed that the X-ray photons are propagating through vacuum before entering and after exiting the phantom. Usually the tissues are surrounded by air, which has a negligible effect on the photon transport.

10.3.1 Free Path Length

All mentioned interaction processes - the photoelectric effect, Compton scattering and Rayleigh scattering - are probabilistic in nature. The according attenuation coefficients μ characterizes the extend of the beam being reduced by its according effect, when passing through the material, in $\mathrm{cm}^2\,\mathrm{g}^{-1}$. For the simulation of photons, the

attenuation coefficients are dependend on the according energy E of the photon and the material at spherical location x of the photon. The coefficients are summarized in Table 10.2.

Interaction Type	Attenuation Coefficient
Photoelectric Effect	$\mu_{\text{PE}}(x, E)$
Rayleigh Scattering	$\mu_{\text{RS}}(x, E)$
Compton Scattering	$\mu_{\text{CS}}(x, E)$

TABLE 10.2: Attenuation coefficients for different photon interaction mechanisms.

The linear attenuation coefficient $\mu(x, E)$ is the sum of the individual attenuation coefficients of the interaction coefficients:

$$\mu(x, E) = \mu_{\text{PE}}(x, E) + \mu_{\text{RS}}(x, E) + \mu_{\text{CS}}(x, E)$$

When an X-ray photon eneters the phantom two main effects are considered:

- **Attenuation:** The photon may be absorbed or Scattered.
- **No Interaction:** The photon may pass through the phantom without any interaction.

The *free path length* t of a photon describes the distance a photon takes to traverse through the phantom before it interacts with matter. Supposing the phantoms location is x and the photon is traveling in the unit direction $\vec{v}$, as in [7] the free path length t follows the distribution given by:

$$t \sim \mu(x, E) \exp\left[-\int_0^t \mu(x + s \cdot \vec{v}) \right]$$

Check: nach dem paper müsste μ abhängig vom Endpunkt sein.

The free path length t is constraint by the maximum distance c ($0 \leq t \leq c$) to the next exit point of the phantom along the ray with direction $\vec{v}$. After leaving the phantom, we assume the photon is reaching the detector or leaving the simulation domain.

The probability of the photon to leave the phantom unaltered is described by the *escape probability* $\mathcal{P}$ of a photon at a given position A with unit direction $\vec{v}$ and energy E as in [7]:

$$\mathcal{P}(x, \vec{v}, E) = \exp\left[-\int_0^c \mu(x + t\vec{v}, E) dt \right],$$

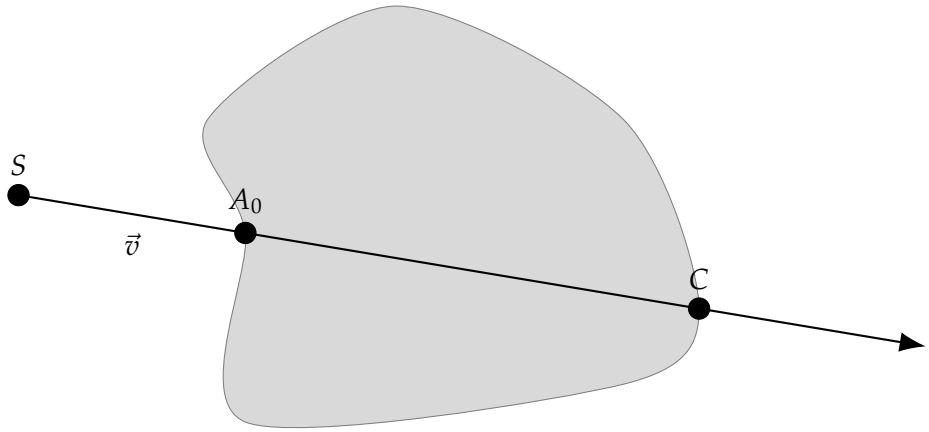


FIGURE 10.8: Illustration of the entry and exit point of the ray of a photon without interaction.

10.3.2 Compton Scattering

Compton scattering is the most dominant interaction mechanism for X-ray photons in tissue [9, Chap. 7]. It occurs when a X-ray photon with considerably higher energy than the binding energy of an outer shell electron collides with this electron. This interaction results in a transfer of energy and momentum. The electron is ejected from the atom, while the incoming photon is scattered at an angle and its energy is reduced.

A portion of the incident photon energy is transferred to the electron, which is referred to as the "recoil electron" or Compton electron. The interaction produces a positive ion, the "recoil electron" and a scattered photon. If the deflection angle of the scattered photon is small, most of the energy is retained by the scattered photon. The deflection angle can vary from 0 to 180 degrees, depending on the energy transfer during the interaction.

For the distribution of the scattering angle θ of the scattered photons in the differential cross section, we apply the formula derived by *Klein and Nishina* in 1928. This equation describes the distribution of the scattering angle of X-ray photons in Compton scattering assuming the electron is initially free and stationary, the relation between energy loss and photon deflection is determined from conservation of momentum and energy between the photon and the recoiling electron [3, 5].

$$\frac{d\sigma}{d\Omega} = \frac{1}{2} r_e^2 \frac{1}{(1+k(1-\cos\theta))^2} \left(1 + \cos^2\theta + \frac{k^2(1-\cos\theta)^2}{1+k(1-\cos\theta)}\right) \tag{10.1}$$

where $k = \frac{e}{mc^2} \approx \frac{E}{0.511}$ with mc^2 being the rest mass energy of the electron, E being the energy of the incident photon and r_e being the classical electron radius.

For sampling from the distribution of the scattering angle we cannot apply the same procedure as for the photon energy in Section 10.2.1, since the Klein-Nishina formula is not a cumulative distribution function (CDF). Instead the method of *rejection sampling* is used.

Rejection sampling is a technique to generate samples from a target distribution by using an envelope distribution that is easy to sample from. The envelope distribution is an upper bound to the original distribution. The basic idea is to sample from this proposal distribution and accept or reject the samples based on a criterion that relates the envelope distribution to the target distribution from Klein-Nishina [11, Chap. 4].

In the context of the distribution of Klein-Nishina, the convergence behavior for $E \to \infty$ and $k \to \infty$ consequently is:

$$f(k,x) \in \mathcal{O}(\frac{1}{k(1-x)}) \tag{10.2}$$

A suitable envelope distribution is of the form

$$h(k,x) = \frac{1}{k(1-x)}$$

with the given conditions

$$\begin{aligned} f(k,1) &= h(k,1) = 2 \\ f(k,-1) &= h(k,-1) = \zeta(k) \\ \zeta(k) &= \frac{2(2k^2+2k+1)}{(2k+1)^3} \end{aligned}$$

By a straightforward calculation, this leads to the following values[14]:

$$\begin{aligned} a(k) &= 2(b(k)-1) \\ b(k) &= \frac{1+\frac{\zeta(k)}{2}}{1-\frac{\zeta(k)}{2}} \end{aligned}$$

Now by using the rejection sampling method, $x = \cos\theta$ is sampled from the distribution h by generating a uniform random variable $r \sim \mathcal{U}(0,1)$ and calculating the value x as follows:

$$x = b(k) - (b(k)-1)\left(\frac{\zeta}{2}\right)^r \tag{10.3}$$

We will accept this value if $\frac{f(k,x)}{h(k,x)} \geq u_1$, where u_1 denotes the random input variable. To derive an exact direction of the scattered photon, we need to further sample an azimuthal angle ϕ, which is done by the second random input variable:

$$\phi = 2\pi u_2 \tag{10.4}$$

Once the scattering angle θ is sampled, the energy of the scattered photon E' can be calculated using the Compton formula:

$$E' = \frac{E}{1+\frac{E}{m_e c^2}(1-\cos\theta)} \tag{10.5}$$

10.3.3 Rayleigh Scattering

Rayleigh scattering is a type of elastic scattering that occurs at low X-ray energies [9, Chap. 7]. The incident photon interacts with several electrons that are usually bound in the outer shells of the atom. In this process, the low energy photon is not ejected, but rather the electrons and in turn thw whole atom is set to vibration with respect to the incident photon's wavelength. The vibrating photon transfers its excess energy to an electromagnetic photon with the same wavelength but probably a different direction than the incident photon. The majority of these scattered photons are emitted in a forward direction. This interaction does not result in the ejection of electrons from the atom and no ionization occurs as no energy is converted into kinetic energy.

Although Rayleigh scattering is taking place in the X-ray tube, it can be excluded from the linear attenuation coefficient, as it does not contribute to the attenuation of the X-ray beam in the same way as Compton scattering or photoelectric absorption. As described in [17], the exclusion of Rayleigh scattering from the linear attenuation coefficient is based on the assumption that the characteristic radiation emitted by the target is isotropic, meaning it is emitted uniformly in all directions. As the X-ray is not attenuated by Rayleigh scattering, it is a common assumption to exclude Rayleigh scattering from the attenuation coefficient.

Chapter 11

The Simulation Algorithm

This chapter presents the algorithm used for X-ray image simulation. It incorporates elements of *Forced Detection* as introduced in [4], in order to accelerate convergence.

Unlike standard Monte Carlo simulations, where a random variable determines whether a photon escapes the phantom or undergoes another scattering event, the forced detection approach employed, calculates the probability of escaping the phantom at each interaction point and reflects this distribution in accordingly updated photon intensities for the case of escaping the phantom and the case of scattering. Instead of probabilistically terminating the photon trajectory, the algorithm tracks it through a predefined number of scattering events N. At each interaction, the remaining intensity is updated to reflect both the probability of Compton scattering and the conditional escape probability. This ensures that both potential outcomes—escape or continued scattering—are implicitly accounted for in the evolving photon intensity. As a result, the simulation maintains physical accuracy while achieving a significant reduction in variance.

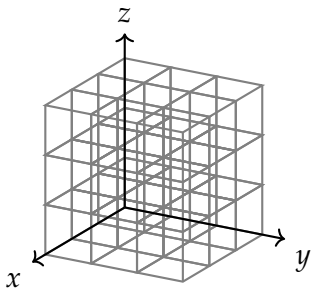
When a photon eventually escapes the phantom, it contributes to the corresponding detector pixel, weighted by its current intensity, photon energy, and escape probability. Conversely, the distance to the next interaction is sampled based on the probability of not escaping the phantom. Further implementation details are discussed in Section 11.1.

This variance reduction strategy enables the simulation to converge more rapidly toward high-resolution images with suppressed noise and well-preserved structural detail.

11.1 Algorithm overview

The algorithm begins by initializing a set of photons, each assigned an initial energy E_0, sampled from the X-ray tube spectrum and an initial direction $\vec{\omega}_0$ sampled uniformly within a cone centered around the principal beam axis. A total of three random variables are drawn for each photon to determine its energy and direction.

Given a fixed source position A representing the location of the X-ray tube, each photon's initial origin is placed within a voxel grid composed of cubic voxels that define the geometry of the scene. Each voxel encodes an integer value identifying a specific material or tissue type. Furthermore, each photon is initialized with an intensity of $W_0 = 1$, which is iteratively updated throughout the simulation to account for attenuation and scattering processes.



Then an algorithm is applied to traverse the photon through the voxel grid until it reaches the first material which is not air. This is done with a ray traversal algorithm. Once the phantom at a point A_0 is reached, the photon transport is simulated by the physical laws described in Chapter 7.

Hereby, as in all other scatter points within the phantom, the free *free path length* t_i is sampled from the exponential distribution based on *Beer-Lambert's law* (Eq. 9.1) and the *total attenuation coefficient* $\mu(A_i, E_i)$ accordingly. A_i hereby denotes the current position and E_i the current energy. First, the *escape probability* depending on the current position A_i and the unit direction $\vec{\omega}_i$ and the energy E_i is computed. The escape probability in Equation 11.1 is the probability of the photon escaping the phantom at the current position A_i without being further scattered.

$$p(A_i, \omega_i, E_i) = \exp\left(- \int\limits_{\overrightarrow{A_i C_i}} \mu(x, E_i) dx \right) \tag{11.1}$$

Hereby C_i denotes the exit point of the phantom in the direction of the photon $\vec{\omega}_i$. The escape probability is used to determine the intensity of the photon without the $(i+1)$-th scatter event.

Using one random variable u_{3i+1}, the free path length is sampled. Hereby both cases are being considered:

1. **Photon escapes the phantom without further scattering:**
   This happens with the portion matching the escape probability $p(A_i, \omega_i, E_i)$. Accordingly
   $$W_{i+1}^{\text{escape}} = W_i \cdot p(A_i, \omega_i, E_i) \tag{11.2}$$
   The photon contributes to the detector signal in direction $\vec{\omega}_i$ with a final intensity of $E_i \cdot W_{i+1}^{\text{escape}}$. In case $i = 0$, this intensity is accounted as priamry intensity.

2. **Photon does not escape the phantom and scatters:**
   This happens with the portion matching the inverse of the escape probability $1 - p(A_i, \omega_i, E_i)$. Accordingly, the free path length is sampled from the exponential distribution by solving Equation 11.3 for t_i:
   $$\exp\left(- \int\limits_0^{t_i} \mu(A_i + s \cdot \vec{\omega}_i, E_i) ds \right) = u_{3i+4} \tag{11.3}$$

Accordingly, the next scatter point is being computed as:

$$A_{i+1} = A_i + t_i \cdot \vec{\omega}_i \tag{11.4}$$

The photon intensity is then updated with:

$$W_{i+1} = W_i \cdot (1 - p(A_i, \omega_i, E_i)) \cdot \frac{\mu_{CS}(A_{i+1}, E_i)}{\mu(A_{i+1}, E_i)} \tag{11.5}$$

In the next step, the photon energy and opening angle after the Compton scatter event is sampled with a second random variable u_{3i+2}. A thrid variable is used to apply a azimuthal rotation around the direction of the photon $\vec{\omega}_i$ to sample the new direction $\vec{\omega}_{i+1}$ of the photon after the scatter event.

This process is repeated according to the maximum scatter order N.

TODO: what happens with the leftover intensity? It is forwarded without any further scattering!

11.2 Sub-Algorithms

To model the relevant physical processes in a structured manner, the main simulation is decomposed into several sub-algorithms. This section describes the purpose and implementation of each sub-algorithm in detail.

11.2.1 Photon Generation

For the photon generation step, two fundamental properties are sampled for each photon:

- *Photon energy*, sampled using a single random variable from the X-ray spectrum.
- *Photon direction*, sampled using two random variables uniformly within a cone defined by the beam axis $\vec{d}$ and opening angle α.

Photon Energy Sampling

The photon energy is sampled from the X-ray tube spectrum, which is represented as a discrete set of energy values with corresponding fluence values. The spectrum was generated using *Spekpy* [16, 18], based on an X-ray tube configured with the parameters listed in Table 11.1.

Parameter	Value
Tube voltage	120 kV
Anode material	Tungsten
Filtration	0.4 mm Tin (Sn), 0.1 mm Copper (Cu)
Target angle	12.5°

TABLE 11.1: Parameters used to generate the X-ray spectrum with *Spekpy*.

Photon energies are sampled using inverse transform sampling. The algorithm normalizes the fluence values to obtain a probability density function (PDF), computes the corresponding cumulative distribution function (CDF) and uses a uniformly distributed random variable to select an energy according to the CDF.

Algorithm 2 Photon Energy Sampling from Spectrum

Require: Array of energies $E = [E_1, E_2, \ldots, E_n]$
Require: Corresponding fluence values $\Phi = [\phi_1, \phi_2, \ldots, \phi_n]$
Require: Number of samples N
Require: Random variable $u \sim \mathcal{U}(0,1)$
Ensure: Sampled photon energies $S = [s_1, s_2, \ldots, s_N]$

// Normalize fluence values:

1: $$T \leftarrow \sum_{i=1}^{n} \phi_i$$

2: $$\mathrm{PDF}[i] \leftarrow \frac{\phi_i}{T}$$

// Compute cumulative distribution function:

3: $$\mathrm{CDF}[1] \leftarrow \mathrm{PDF}[1]$$

4: **for** $i = 2$ to n **do**
5: $\quad \mathrm{CDF}[i] \leftarrow \mathrm{CDF}[i-1] + \mathrm{PDF}[i]$
6: **end for**

// Note: $PDF[i] > 0$ in the spectrum, therefore CDF is strictly increasing.

7: Create interpolating function $\mathrm{InverseCDF}(u)$ from $(\mathrm{CDF}[i], E[i])$
8: **return** $\mathrm{InverseCDF}(u)$

Photon Direction Sampling

The direction of each photon is sampled uniformly within a cone defined by the beam axis $\vec{d}$ and opening angle α. This requires two independent random variables $u_1, u_2 \sim \mathcal{U}(0,1)$.

Algorithm 3, adapted from [21], generates a random unit vector $\vec{v}$ uniformly distributed within a cone of half-angle α around the direction $\vec{d}$:

1. **Sampling the polar angle** θ:
   Draw $u_1 \sim \mathcal{U}(0,1)$ and compute

   $$\theta = \arccos\left(1 - u_1(1 - \cos\alpha)\right).$$

   This corresponds to inverse transform sampling such that $\cos\theta$ is uniformly distributed on $[\cos\alpha, 1]$, resulting in uniform sampling over the spherical cap.

2. **Sampling the azimuthal angle** ϕ:
   Draw $u_2 \sim \mathcal{U}(0,1)$ and compute

   $$\phi = 2\pi u_2,$$

   ensuring a uniform distribution around the cone axis.

3. **Constructing the local direction vector**:
   In a local spherical coordinate system with the cone axis aligned along the z-axis, the sampled unit vector is

$$\vec{v}_{\text{local}} = \begin{pmatrix} \sin\theta\cos\phi \\ \sin\theta\sin\phi \\ \cos\theta \end{pmatrix}.$$

4. **Rotation into global coordinates**:
   To align the cone axis from the local z-axis to an arbitrary unit vector $\vec{d}$, an orthonormal basis $(\vec{u}, \vec{v}, \vec{d})$ is constructed via the Gram–Schmidt process:
   - Choose a helper vector $\vec{a} = (0,0,1)$ if $|d_3| < 0.999$, otherwise $\vec{a} = (1,0,0)$.
   - Compute $\vec{u} = \frac{\vec{a}\times\vec{d}}{\|\vec{a}\times\vec{d}\|}$.
   - Set $\vec{v} = \vec{d} \times \vec{u}$ to complete a right-handed orthonormal basis.

   Finally, rotate $\vec{v}_{\text{local}}$ into global coordinates via the transformation

$$\vec{v}_{\text{global}} = (\vec{v}_{\text{local}})_x\vec{u} + (\vec{v}_{\text{local}})_y\vec{v} + (\vec{v}_{\text{local}})_z\vec{d}.$$

Algorithm 3 Uniform Direction Sampling Within a Cone

Require: Cone angle α
Require: Unit beam direction vector $\vec{d} = (d_1, d_2, d_3)$
Require: Random variables $u_1, u_2 \sim \mathcal{U}(0,1)$
Ensure: Sampled direction vector $\vec{v}$ uniformly within cone around $\vec{d}$

// Calculate angles according to samples
$\theta \leftarrow \arccos(1 - u_1(1 - \cos\alpha))$ // Polar angle
$\phi \leftarrow 2\pi u_2$ // Azimuthal angle
// Calculate local direction vector in spherical coordinates

$$\vec{v}_{\text{local}} \leftarrow \begin{pmatrix} \sin\theta\cos\phi \\ \sin\theta\sin\phi \\ \cos\theta \end{pmatrix}$$

// Orthonormal basis construction (Gram-Schmidt)
if $|d_3| < 0.999$ **then**
    $\vec{a} \leftarrow (0,0,1)$
else
    $\vec{a} \leftarrow (1,0,0)$
end if
$\vec{u} \leftarrow \frac{\vec{a}\times\vec{d}}{\|\vec{a}\times\vec{d}\|}$ // Orthogonal vector
$\vec{v} \leftarrow \vec{d} \times \vec{u}$ // Complete right-handed basis
// Rotate local vector into global coordinates
$\vec{v}_{\text{global}} \leftarrow \vec{u}(\vec{v}_{\text{local}})_x + \vec{v}(\vec{v}_{\text{local}})_y + \vec{d}(\vec{v}_{\text{local}})_z$
return $\vec{v}_{\text{global}}$

11.2.2 Ray Traversal

The ray traversal algorithm is responsible for simulating the propagation of a photon through the voxel grid. It is used to forward the photon until it reaches the first chemical compound, which is not air (or is air). It is also used to simulate the photon transport within the phantom until it reaches the exit point of the phantom.

The Input values of the algorithm are:

- The voxel grid *materialGrid* representing the geometry of the scene.
- The initial position of the photon A.
- The (unit) direction of the photon $\vec{\omega}$.

The algorithm traverses the photon through the voxel grid, voxel by voxel while checking the material of the next voxel and iterrupts in case the next voxel is not air (or is air).

The return values of the algorithm are:

- The coordinates of the final postion: C
- An array of all crossed voxel indices: *crossedVoxels*
- An array of all crossed voxel materials: *crossedMaterials*
- An array of al entry points of each crossed voxel: *entryPoints*
- An array of the distances traveres in each voxel: *distances*
- A boolean mask indicating whether the photon exited the grid: *exitGrid*

The sequence of the algorithm can be summarized as follows:

I. Initialize empty arrays for *crossedVoxels*, *crossedMaterials*, *entryPoints* and *distances*.

II. Convert the photon position A into voxel coordinates *pos*.

III. Determine the indices of the next voxel *voxelIdx* based on the *pos* and the direction $\vec{\omega}$.

IV. Set *exitGrid* to false.

V. **If** the *voxelIdx* of the next voxel is out of bounds, set *exitGrid* to true and exit the algorithm.

VI. Determine *material* of the next *voxelIdx*

VII. **If** the *material* is not air, exit the algorithm.

VIII. **While** *material* is air:

1. Append *voxelIdx* to *crossedVoxels*.
2. Append *material* to *crossedMaterials*.
3. Append *pos* to *entryPoints*.
4. Determine *maxDist* to the next voxel boundary.
5. Append *maxDist* to *distances*.

6. Update *pos* by adding $\vec{\omega} \cdot maxDist$.
7. Determine $voxelIdx$ of the next voxel based on the updated *pos*.
8. **If** the $voxelIdx$ is out of bounds, set $exitGrid$ to true and exit the algorithm.
9. Determine *material* of the next $voxelIdx$.

When the algorithm finishes, the final Position *C* is set to the last value of *position*.

To use this agorithm for travering the photons through air until they reach the phantom and to traverse the photons through the phantom until they reach the exit point of the phantom, the algorithm is called with the boolean value $throughAir$. In case $throughAir = true$, the algorithm will traverse the photons through air until they reach the first material which is not air. In case $throughAir = false$, the algorithm will traverse the photons through the phantom until they reach the exit point of the phantom.

Therefore we define a short helper function in Algorithm 4 beforehand, which generates the specific expression depending on the boolean value of $throughAir$.

Algorithm 4 Ray Traversal Helper Function

Require: Boolean $throughAir$
Require: *material*
Ensure: Boolean value
1: **if** $throughAir$ **then**
2: **return** *material* = air
3: **else**
4: **return** *material* $\neq$ air
5: **end if**

Algorithm 5 implements the process of ray traversing through the voxel grid in detail.

Algorithm 5 Ray Traversal Algorithm

Require: Boolean *throughAir*
Require: Material grid $materialGrid \in \mathbb{Z}^{X \times Y \times Z}$, voxel size $s \in \mathbb{R}^+$
Require: Initial pos $A \in \mathbb{R}^3$, direction $\vec{\omega} \in \mathbb{R}^3$
Ensure: Final pos $C \in \mathbb{R}^3$
Ensure: Arrays *crossedVoxels*, *crossedMaterials*, *entryPoints*, *distances*
Ensure: Boolean *exitGrid* indicating whether the photon exited the grid

$pos \leftarrow O/s$
$exitGrid \leftarrow$ false
$currentVoxelIdx \leftarrow \lfloor pos \rfloor$
$negDir \leftarrow \vec{\omega} < 0$
$onB \leftarrow (pos = \lfloor currentVoxelIdx \rfloor)$
$nextVoxelIdx \leftarrow currentVoxelIdx$
$nextVoxelIdx[negDir \wedge onB] \leftarrow nextVoxelIdx[negDir \wedge onB] - 1$
if any($nextVoxelIdx < 0$) or any($nextVoxelIdx \geq$ shape($materialGrid$)) **then**
    $exitGrid \leftarrow$ true
    $C \leftarrow pos$
    return $C, crossedVoxels, crossedMaterials, entryPoints, distances, exitGrid$
end if
$material \leftarrow materialGrid[nextVoxelIdx]$
while Algorithm 4(throughAir, material) **do**
    $crossedVoxels$.append($nextVoxelIdx$)
    $crossedMaterials$.append($material$)
    $entryPoints$.append(pos)
    $fracPos \leftarrow pos - \lfloor pos \rfloor$
    $maxDist \leftarrow [\infty, \infty, \infty]$
    $negDir \leftarrow \vec{\omega} < 0$
    $posDir \leftarrow \vec{\omega} > 0$
    $onB \leftarrow (pos = \lfloor currentVoxelIdx \rfloor)$
    $maxDist[negDir \wedge onB] \leftarrow -1/\vec{\omega}[negDir \wedge onB]$
    $maxDist[negDir \wedge \neg onB] \leftarrow -fracPos[negDir \wedge \neg onB]/\vec{\omega}[negDir \wedge \neg onB]$
    $maxDist[posDir] \leftarrow (1 - fracPos[posDir])/\vec{\omega}[posDir]$
    $distance = \min(maxDist)$
    $distances$.append($maxDist$)
    $pos \leftarrow pos + \vec{\omega} \cdot distance$
    $currentVoxelIdx \leftarrow \lfloor pos \rfloor$
    $nextVoxelIdx \leftarrow currentVoxelIdx$
    $nextVoxelIdx[negDir \wedge onB] \leftarrow nextVoxelIdx[negDir \wedge onB] - 1$
    if any($nextVoxelIdx < 0$) or any($nextVoxelIdx \geq$ shape($materialGrid$)) **then**
        $exitGrid \leftarrow$ true
        $C \leftarrow pos$
        return $C, crossedVoxels, crossedMaterials, entryPoints, distances, exitGrid$
    end if
    $material \leftarrow materialGrid[nextVoxelIdx]$
end while
$C \leftarrow pos$
return $C, crossedVoxels, crossedMaterials, entryPoints, distances, exitGrid$

11.2.3 Forced Detection

The forced detection algorithm is responsible to apply one iteration of the forced detection process to a photon within the phantom. Hereby the algorithm accounts:

- The voxel grid *materialGrid* representing the geometry of the scene.
- The voxel grid *totalAttenuationGrid* representing the attenuation coefficients of the materials in the voxel grid *materialGrid*.
- The voxel grid *comptonAttenuationGrid* representing the Compton scattering coefficients of the materials in the voxel grid *materialGrid*.
- The voxel grid *absorptionGrid* representing the absorption coefficients of the materials in the voxel grid *materialGrid*.
- The initial position of the photon A_i.
- The (unit) direction of the photon $\vec{\omega}_i$.
- The initial energy of the photon E_i.
- The initial intensity of the photon W_i.
- A random variable $u \sim \mathcal{U}(0,1)$ to sample the free path length.
- The voxel size s.

The algorithm then applies the ray traversal algorithm (Algorithm 5) to traverse the photon through the phantom until it reaches the exit point of the phantom C_i. Obtaining the exit point C_i, the arrays of *crossedVoxels* and *distances*, now the attenuation coefficients of the materials along the ray can be extracted from the voxel grids for the crossed voxels:

$$\begin{aligned} totalAttenuationCoefficients &= totalAttenuationGrid[crossedVoxels] \\ comptonScatteringCoefficients &= comptonAttenuationGrid[crossedVoxels] \end{aligned}$$

With the following helper algorithm (Algorithm 6), the *escapeProbability* is computed. Further the last distance index k and last interpolation factor f are returned, to solve Equation 11.3 for the free path length t_i, which can then be simply computed by:

$$t_i = \sum_{j=0}^{k-1} distances[j] + f \tag{11.6}$$

Now the next scatter point $A_{i+1} = A_i + t_i \cdot s \cdot \vec{\omega}_i$ can be determined and the new intensities together with the exit point C_i can be computed:

Algorithm 6 Partial Product Sum for Free Path Sampling in Forced Detection

Require: Arrays $distances, totalAttenuationCoefficients \in \mathbb{R}^n$, weight $u \in [0, 1]$
Ensure: Last distance index k, last interpolation factor f, $escapeProbability$

$S_{\text{total}} \leftarrow 0$
Initialize array $P[0 \dots n-1]$
for $i \leftarrow 0$ to $n-1$ **do**
    $P[i] \leftarrow distances[i] \cdot totalAttenuationCoefficients[i]$
    $S_{\text{total}} \leftarrow S_{\text{total}} + P[i]$
end for
$T \leftarrow u \cdot S_{\text{total}}$
$S \leftarrow 0$
for $i \leftarrow 0$ to $n-1$ **do**
    if $S + P[i] > T$ **then**
        $f \leftarrow \frac{T-S}{totalAttenuationCoefficients[i]}$
        return (i, f, S_{total})
    end if
    $S \leftarrow S + P[i]$
end for
return $(n, 1.0, S_{\text{total}})$ // u = 1.0 or exact fit

$$
\begin{aligned}
voxelIdx &= \lfloor A_{i+1}/s \rfloor \\
W_{i+1} &= W_i \cdot (1 - escapeProbability) \cdot \frac{comptonAttenuationGrid[voxelIdx]}{totalAttenuationCoefficients[voxelIdx]} \\
W_{i+1}^{\text{escape}} &= W_i \cdot escapeProbability
\end{aligned}
$$

And accordingly the forced detection algorithm returns the following values:

- The new position of the photon A_{i+1}.
- The new intensity of the photon after the scatter event W_{i+1}.
- The intensity of the photon for the case of escaping the phantom W_{i+1}^{escape}.
- The exit point of the phantom C_i.
- The Compton scattering coefficient μ_{CS} at the scatter point A_{i+1}.

Algorithm 7 Forced Detection Algorithm

Require: Material grid $materialGrid \in \mathbb{Z}^{X \times Y \times Z}$
Require: Total attenuation grid $totalAttenuationGrid \in \mathbb{R}^{X \times Y \times Z}$
Require: Compton scattering grid $comptonAttenuationGrid \in \mathbb{R}^{X \times Y \times Z}$
Require: Absorption grid $absorptionGrid \in \mathbb{R}^{X \times Y \times Z}$
Require: Initial position $A_i \in \mathbb{R}^3$
Require: Unit direction $\vec{\omega}_i \in \mathbb{R}^3$
Require: Initial energy $E_i \in \mathbb{R}^+$
Require: Initial intensity $W_i \in \mathbb{R}^+$
Require: Random variable $u \sim \mathcal{U}(0,1)$
Require: Voxel size $s \in \mathbb{R}^+$
Ensure: New position $A_{i+1} \in \mathbb{R}^3$
Ensure: New intensity $W_{i+1} \in \mathbb{R}^+$
Ensure: Escape intensity $W_{i+1}^{\text{escape}} \in \mathbb{R}^+$
Ensure: Exit point $C_i \in \mathbb{R}^3$
Ensure: Compton Attenuation coefficient μ_{CS}

$C_i, crossedVoxels, crossedMaterials, entryPoints, distances, exitGrid \leftarrow$ Algorithm 5$(throughAir = true, materialGrid, s, A_i)$
if exitGrid **then**
    return $(C_i, W_i, 0, C_i)$ // Photon escaped the phantom
end if
$totalAttenuationCoefficients \leftarrow totalAttenuationGrid[crossedVoxels]$
$comptonScatteringCoefficients \leftarrow comptonAttenuationGrid[crossedVoxels]$
$absorptionCoefficients \leftarrow absorptionGrid[crossedVoxels]$
$k, f, escapeProbability \leftarrow$ Algorithm 6$(distances, totalAttenuationCoefficients, u)$
$escapeProbability \leftarrow \frac{escapeProbability}{\sum_{j=0}^{k-1} distances[j] \cdot totalAttenuationCoefficients[j]}$
$t_i \leftarrow \sum_{j=0}^{k-1} distances[j] + f$
$A_{i+1} \leftarrow A_i + t_i \cdot s \cdot \vec{\omega}_i$
$voxelIdx \leftarrow \lfloor A_{i+1}/s \rfloor$
$\mu_{\text{CS}} \leftarrow comptonScatteringCoefficients[voxelIdx]$
$W_{i+1} \leftarrow W_i \cdot (1 - escapeProbability) \cdot \frac{\mu_{\text{CS}}}{totalAttenuationCoefficients[voxelIdx]}$
$W_{i+1}^{\text{escape}} \leftarrow W_i \cdot escapeProbability$
return $(A_{i+1}, W_{i+1}, W_{i+1}^{\text{escape}}, C_i, \mu_{\text{CS}})$

11.2.4 Photon Exit Point Determination

To determine the exit point of the photon in the world after exiting the phantom, a more efficient algorithm than the ray traversal algorithm from Section 11.2.2 is used can be applied. This algorithm yields the exit point of the phantom which is used to determin the detector pixel the photon contributes to.

The algorithm is initialized with the following parameters:

- The voxel grid shape (N_x, N_y, N_z) of the *materialGrid* representing the geometry of the scene.
- The initial position of the photon C^{phantom}.
- The (unit) direction of the photon $\vec{\omega}$.

- The voxel size s.

The algorithm then computes the point where the photon exits the voxel grid efficiently and returns the exit point C^{grid} and the according Coordinates $C^{\text{gridCoords}}$. The algorithm is implemented as follows:

Algorithm 8 Compute Exit Point of Ray from Voxel Grid

Require: Ray origin $C^{\text{phantom}} \in \mathbb{R}^3$, direction $\vec{\omega} \in \mathbb{R}^3$
Require: Grid shape (N_x, N_y, N_z), voxel size $s \in \mathbb{R}^+$
Ensure: Exit point C^{grid}, $C^{\text{gridCoords}}$

$C^{\text{phantomCoords}} \leftarrow C^{\text{phantom}} / s$ // Convert to voxel coordinates
$x_{\min} \leftarrow 0, x_{\max} \leftarrow N_x$
$y_{\min} \leftarrow 0, y_{\max} \leftarrow N_y$
$z_{\min} \leftarrow 0, z_{\max} \leftarrow N_z$
for axis in {x, y, z} **do**
    $o \leftarrow C^{\text{phantomCoords}}_{\text{axis}}$
    $d \leftarrow \vec{\omega}_{\text{axis}}$
    if $d > 0$ **then**
        $t^{(\text{axis})} \leftarrow \frac{x_{\max} - o}{d}$
    else if $d < 0$ **then**
        $t^{(\text{axis})} \leftarrow \frac{x_{\min} - o}{d}$
    else if $d = 0$ **then**
        $t^{(\text{axis})} \leftarrow -\infty$
    end if
end for
$t_{\text{exit}} \leftarrow \min(t^{(x)}, t^{(y)}, t^{(z)})$ // Exit time
$C^{\text{gridCoords}} \leftarrow C^{\text{phantomCoords}} + t_{\text{exit}} \cdot \vec{\omega}$
$C^{\text{grid}} \leftarrow C^{\text{gridCoords}} \cdot s \cdot \vec{\omega}$
return C^{grid}, $C^{\text{gridCoords}}$

11.2.5 Compton Scattering

The Compton scattering algorithm is responsible for simulating the physical process for the event of Compton scattering. Based on the Klein-Nishina formula (Equation 10.1) and the rejection sampling method from Section 10.3.2, the algorithm samples the scatter angle and computes the new photon energy and direction after the scattering event.

The algorithm initializes with the following parameters:

- The initial energy of the photon E_i.
- The (unit) direction of the photon $\vec{\omega}_i$.
- Two random variables $u_1, u_2 \sim \mathcal{U}(0, 1)$ to sample the new photon energy and direction.

The algorithm applies the Klein-Nishina procedure to compute the scatter angle and therefore utilizes the electron rest energy E_{rest} and follow this procedure:

1. Initialize

$$k = E_i/mc^2, \qquad \zeta(k) = \frac{2(2k^2+2k+1)}{(2k+1)^3}, \qquad b(k) = \frac{1+\frac{\zeta}{2}}{1-\frac{\zeta}{2}}, \qquad a(k) = 2(b-1).$$

2. Rejection Sampling loop:
   2.1. Generate random number $r \sim \mathcal{U}(0,1)$.
   2.2. Calculate

$$x = b - (b+1)\left(\frac{\zeta}{2}\right)^r$$
$$f(k,x) = \frac{1+x^2+\frac{k^2(1-x)^2}{1+k(1-x)}}{(1+k(1-x))^2}$$
$$h(k,x) = \frac{a}{b-x}$$

   2.3. **If** $u_1 \leq \frac{f(k,x)}{h(k,x)}$ **then**: $\theta = \arccos(x)$

3. Calculate photon energy with Equation 10.5 and direction using the azimuthal angle ϕ from Equation 10.4.

Given the scatter angle θ, the new photon energy E_{i+1} is computed as in [12]:

$$E_{i+1} = \frac{E_i}{1+k(1-\cos\theta)}. \tag{11.7}$$

With the random variable u_2 an azimuthal angle ϕ is sampled to determine the new direction.

$$\phi = 2\pi u_2 \tag{11.8}$$

Assuming the vector $\vec{u}$ is an orthogonal unit vector $\vec{u} \perp \vec{\omega}_i$ and $\vec{v} = \vec{u} \times \vec{\omega}_i$, then the new direction $\vec{\omega}_{i+1}$ is as follows:

$$\vec{\omega}_{i+1} = \sin(\theta)cos(\phi) \cdot u + \sin(\theta)sin(\phi) \cdot v + \cos(\theta) \cdot \vec{\omega}_i \tag{11.9}$$

This leads to the following return values of the algorithm:

- The new photon energy E_{i+1} after the Compton scattering event.
- The new (unit) direction $\vec{\omega}_{i+1}$ of the photon after the Compton scattering event.

In the pseudoalgorithm is more explicitly describing this process in detail in Algorithm 9 by implementing omitted tweaks and details.

Algorithm 9 Compton Scattering Algorithm

Require: Initial photon energy E_i, direction $\vec{\omega}_i$
Require: Electron rest energy E_{rest}
Require: Random variables $u_1, u_2 \sim \mathcal{U}(0,1)$
Ensure: New photon energy E_{i+1}, direction $\vec{\omega}_{i+1}$

// Scatter angle sampling

$k \leftarrow E_i / E_{\text{rest}}$
$\varepsilon_0 \leftarrow 1/(2k+1)$
repeat
  Generate $r \sim \mathcal{U}(0,1)$
  if $r < 0.5$ **then**
    $\varepsilon \leftarrow \varepsilon_0 + (1-\varepsilon_0) \cdot 2r$
  else
    $\varepsilon \leftarrow \varepsilon_0 + (1-\varepsilon_0) \cdot 2(1-r)$
  end if
  $\cos\theta \leftarrow 1 + \frac{1}{k}\left(1 - \frac{1}{\varepsilon}\right)$
  if $|\cos\theta| \leq 1$ **then**
    $\sin^2\theta \leftarrow 1 - \cos^2\theta$
    if $u_1 \leq \frac{1}{2}\left(\varepsilon + \frac{1}{\varepsilon} - \sin^2\theta\right)$ **then**
      $\theta \leftarrow \arccos(\cos\theta)$
      break
    end if
  end if
until accepted

// New photon energy calculation

$E_{i+1} \leftarrow \frac{E_i}{1+k(1-\cos\theta)}$

// Orthonormal basis construction

$\phi \leftarrow 2\pi u_2$
if $|(\vec{\omega}_i)_z| < 0.999$ **then**
  $\vec{a} \leftarrow (0,0,1)$
else
  $\vec{a} \leftarrow (1,0,0)$
end if
$\vec{u} \leftarrow \frac{\vec{a} \times \vec{\omega}_i}{\|\vec{a} \times \vec{\omega}_i\|}$
$\vec{v} \leftarrow \vec{\omega}_i \times \vec{u}$

// New direction calculation

$\vec{\omega}_{i+1} \leftarrow \sin\theta\cos\phi \cdot \vec{u} + \sin\theta\sin\phi \cdot \vec{v} + \cos\theta \cdot \vec{\omega}_i$
return $E_{i+1}, \vec{\omega}_{i+1}$

11.3 Algorithm Composition

The main algorithm is composing the sub-algorithms from Section 11.2.

The algorithm is initialized with the following parameters:

- The voxel grid *materialGrid* representing the geometry of the scene.
- The voxel grid *totalAttenuationGrid* representing the total attenuation coefficients of the materials in the voxel grid *materialGrid*.

- The voxel grid *comptonAttenuationGrid* representing the Compton scattering coefficients of the materials in the voxel grid *materialGrid*.
- The voxel grid *absorptionGrid* representing the absorption coefficients of the materials in the voxel grid *materialGrid*.
- The source position S of the X-ray tube.
- The number of scatter events N to simulate.
- The opening angle α of the X-ray beam.
- The voxel size s of the voxel grid.
- A sequence of random variables $u \sim \mathcal{U}(0,1)^{3(N+1)}$ to sample the photon energies, directions and free path lengths.

Hereby, the photon generation algorithm is called to generate the photon energies and directions.

Then the photon is initialized with the an initial energy E_0, an initial direction $\vec{\omega}_0$ and an initial intensity $W_0 = 1$ by applying Algorithm 2 and Algorithm 3. Based on the initial position $A_0 = S$, direction $\vec{\omega}$, the voxel grid and voxel size s, the ray traversal algorithm (Algorithm 5) is applied to traverse the photon through the voxel grid until it reaches the first material which is not air. The exit point of the phantom is denoted as C_0.

The loop over the scatter events:

Later N iterations of the forced detection algorithm (Algorithm 7) are applied together with one random variable to simulate the photon transport through the phantom. In each iteration, the photon position and intensity are updated. With Algorithm 8, the exit point C_i^{grid} of the photon is computed and captured together with the relevant intensity $E_i \cdot W_{i+1}^{\text{escape}}$.

Then the Compton scatter event is simulated, taking into account the photon energy E_i, the direction $\vec{\omega}_i$ and the Compton scattering coefficient μ_{CS} at A_{i+1}. Together with two random variables, the new photon energy E_{i+1} with an according direction $\vec{\omega}_{i+1}$ is sampled.

Algorithm 10 Main Simulation Algorithm Composition

Require: Material grid *materialGrid*
Require: Total attenuation grid *totalAttenuationGrid*
Require: Compton attenuation grid *comptonAttenuationGrid*
Require: Absorption grid *absorptionGrid*
Require: Source position S
Require: Number of scatter events N
Require: Beam opening angle α
Require: Voxel size s
Require: Random variables $u \sim \mathcal{U}(0,1)^{3(N+1)}$
Ensure: Detector contributions (primary and scatter signals)

// Photon generation
$E_0 \leftarrow$ Algorithm 2(spectrum, u_1)
$\vec{\omega}_0 \leftarrow$ Algorithm 3(α, beam axis, u_2, u_3)
$W_0 \leftarrow 1$
// Traverse photon through air to phantom
$A_0, \ldots \leftarrow$ Algorithm 5(throughAir=true, *materialGrid*, s, S, $\vec{\omega}_0$)
for $i = 0$ to $N-1$ **do**
// Forced detection step
$A_{i+1}, W_{i+1}, W_i^{\text{escape}}, C_i^{\text{phantom}}, \mu_{\text{CS}} \leftarrow$ Algorithm 7(*materialGrid*, *totalAttenuationGrid*, *comptonAttenuationGrid*, *absorptionGrid*, A_i, $\vec{\omega}_i$, E_i, W_i, u_{3i+1}, s)
// Record escaped photon contribution
if $W_i^{\text{escape}} > 0$ **then**
$C_i^{\text{world}}, \ldots \leftarrow$ Algorithm 8(C_i^{phantom}, $\vec{\omega}_i$, grid shape, s)
Add $E \cdot W_i^{\text{escape}}$ to detector pixel at C_i^{world}
end if
// Compton scatter step
$E_{i+1}, \vec{\omega}_{i+1} \leftarrow$ Algorithm 9(E, $\vec{\omega}_i$, μ_{CS}, u_{3i+2}, u_{3i+3})
end for
// Handle leftover intensity
$C_N^{\text{world}}, \ldots \leftarrow$ Algorithm 8(A_N, $\vec{\omega}_N$, grid shape, s)
Add $E \cdot W$ to detector pixel intensity at C_N^{world}

Bibliography

[1] George Cybenko. "Approximation by superpositions of a sigmoidal function". In: *Mathematics of control, signals and systems* 2.4 (1989), pp. 303–314.

[2] Bastien Doignies. "Echantillonnage différentiable et simulations Monte Carlo". PhD thesis. Université Claude Bernard-Lyon I, 2024.

[3] J Hubbell. "Photon cross sections, attenuation coefficients and energy absorption coefficients". In: *National Bureau of Standards Report NSRDS-NBS29, Washington DC* (1969).

[4] H.W.A.M. de Jong, E.T.P. Slijpen, and F.J. Beekman. "Acceleration of Monte Carlo SPECT simulation using convolution-based forced detection". In: *IEEE Transactions on Nuclear Science* 48.1 (2001), pp. 58–64. DOI: 10.1109/23.910833.

[5] Oskar Klein and Yoshio Nishina. "The scattering of light by free electrons according to Dirac's new relativistic dynamics". In: *Nature* 122.3072 (1928), pp. 398–399.

[6] Gunther Leobacher and Friedrich Pillichshammer. *Introduction to quasi-Monte Carlo integration and applications*. Springer, 2014.

[7] Guiyuan Lin, Shiwo Deng, and Xiaoqun Wang. "An efficient quasi-Monte Carlo method with forced fixed detection for photon scatter simulation in CT". In: *PLOS ONE* 18 (Aug. 2023), e0290266. DOI: 10.1371/journal.pone.0290266.

[8] Guiyuan Lin et al. "Scatter correction based on quasi-Monte Carlo for CT reconstruction". In: *arXiv preprint arXiv:2501.05039* (2025).

[9] *Medical Imaging Systems : An Introductory Guide*. eng. Image Processing, Computer Vision, Pattern Recognition, and Graphics; 11111. Cham, 2018. Chap. 7,8. ISBN: 9783319965208.

[10] Siddhartha Mishra and T Konstantin Rusch. "Enhancing accuracy of deep learning algorithms by training with low-discrepancy sequences". In: *SIAM Journal on Numerical Analysis* 59.3 (2021), pp. 1811–1834.

[11] Thomas Müller-Gronbach, Erich Novak, and Klaus Ritter. *Monte Carlo-Algorithmen*. Springer-Verlag, 2012.

[12] Dylan R Nelson. "COMPTON SCATTERING". In: (2007). URL: https://www.ita.uni-heidelberg.de/~dnelson/storage/ucb.phy111.spr2007/dnelson.com.pdf.

[13] Art B Owen. "Multidimensional variation for quasi-Monte Carlo". In: *International Conference on Statistics in honour of Professor Kai-Tai Fang's 65th birthday*. World Scientific. 2005, pp. 49–74.

[14] Emin N Özmutlu. "Sampling of angular distribution in Compton scattering". In: *International journal of radiation applications and instrumentation. Part A. Applied radiation and isotopes* 43.6 (1992), pp. 713–715.

[15] Friedrich Pillichshammer. "Zahlentheoretische Methoden in der Numerik". In: *Vorlesungsskript, Johannes Kepler Universität Linz* (2010).

[16] Gavin Poludniowski. *SpekPy: Python package for simulating X-ray spectra*. Version 2.0.13. 8.08.2024. URL: https://bitbucket.org/caxtus/book/src/master/.

[17] Gavin Poludniowski, Artur Omar, and Pedro Andreo. *Calculating X-ray Tube Spectra: Analytical and Monte Carlo Approaches*. CRC Press, 2022.

[18] Gavin Poludniowski et al. "SpekPy v2. 0—a software toolkit for modeling x-ray tube spectra". In: *Medical Physics* 48.7 (2021), pp. 3630–3637.

[19] A. Sisniega et al. "Automatic Monte-Carlo Based Scatter Correction For X-ray cone-beam CT using general purpose graphic processing units (GP-GPU): A feasibility study". In: *2011 IEEE Nuclear Science Symposium Conference Record*. 2011, pp. 3705–3709. DOI: 10.1109/NSSMIC.2011.6153699.

[20] Jörg Steidel et al. "Dose reduction potential in diagnostic single energy CT through patient-specific prefilters and a wider range of tube voltages". In: *Medical physics* 49.1 (2022), pp. 93–106.

[21] Murugesan Venkatapathi et al. "An O (n) algorithm for generating uniform random vectors in n-dimensional cones". In: *arXiv preprint arXiv:2101.00936* (2021).