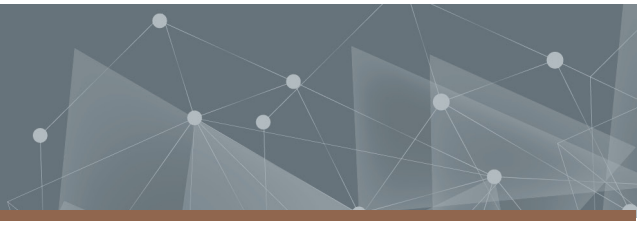
CHALMERS
UNIVERSITY OF TECHNOLOGY



Accelerating computations for dark matter direct detection experiments via neural networks and GPUs

Master's thesis in Physics

HANNA OLVHAMMAR

DEPARTMENT OF PHYSICS

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2023
www.chalmers.se

Master's thesis 2023

Accelerating computations for dark matter direct detection experiments via neural networks and GPUs

HANNA OLVHAMMAR

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Physics
Division of Subatomic, High Energy and Plasma Physics
Chalmers University of Technology
Gothenburg, Sweden 2023

Accelerating computations for dark matter direct detection experiments via neural networks and GPUs
HANNA OLVHAMMAR

Supervisors: Riccardo Catena and Einar Urdshals
Examiner: Riccardo Catena

Master's Thesis 2023
Department of Physics
Division of Subatomic, High Energy and Plasma Physics
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2023

Accelerating computations for dark matter direct detection experiments via neural networks and GPUs

HANNA OLVHAMMAR
Department of Physics
Chalmers University of Technology

Abstract

There is indisputable evidence for the existence of dark matter (DM). Examples are the rotation curves of galaxies, the velocity dispersions of galaxy clusters and dark matter density measurements. One of the biggest questions in physics today concerns the nature of dark matter, and the most promising theory is that dark matter consists of one or more new particle species. To discover the nature of dark matter particles, they need to be inferred from collider experiments or found via indirect or direct detection. Since none of these alternatives has led to conclusive results within the current theoretical frameworks, new approaches should be investigated. In this thesis, sub-GeV dark matter particles are studied through interactions in direct detection experiments described with an effective field theory (EFT). More specifically, dark matter-induced electronic transition rates in crystal detectors are studied. The rate of electronic transitions is described with EFT scattering amplitudes, which introduce many model-independent coupling strengths. By computing transition rates corresponding to different sets of EFT parameters, direct detection data can be used for inferring properties of dark matter particles without relying on any specific theoretical framework. Since the computation of the electronic transition rates is very expensive, the aim of this thesis is to implement a deep neural network for fast predictions of transition rates. Furthermore, since the neural network requires a large data set for training, the generation of training data was accelerated using computations on graphics processing units (GPUs). I developed two neural networks, one with the DM mass as input and one with the DM mass and two EFT coupling strengths as inputs, that are about 600 times faster than the original computations and capture the overall behaviour of the transition rates. However, the relative error of the predictions has a standard deviation of about 30% with a mean of around 0%. On the other hand, the GPU computations are about 16 times faster than the original computations and have negligible error while being able to compute transition rates corresponding to all 28 coupling strengths. I conclude that there is great potential for using both neural networks and GPUs for dark matter research, and suggest further improvements.

Keywords: Dark matter, transition rate, direct detection, machine learning, neural networks, GPU, CUDA, parallelisation, crystal detector

Acknowledgements

I would like to thank my supervisors, Riccardo and Einar, for trusting me with this interesting project. You always showed confidence in my ability to work independently and encouraged me to implement new ideas, even when they diverged from the original plan.

I would also like to thank my family and friends, who have supported me unconditionally not only during my work on this thesis but throughout the last five years.

Hanna Olvhammar, Gothenburg, June 2023

Contents

1
Introduction

While the nature of dark matter (DM) is unknown, there is strong experimental evidence for its existence. This chapter provides an introduction to the subject of dark matter research, both in terms of evidence and current experimental methods. Furthermore, the aim and outline of the thesis are specified.

1.1 Background

Immense contributions to physics were made in the 20th century, resulting in theories such as quantum physics, the standard model and general relativity. While these theories have become well-established and explain a wide range of observed phenomena, several questions remain about the nature of our universe. One concerns the massive, invisible substance thoroughly confirmed by experiments: What is dark matter?

When measuring the dynamical properties of galaxies, it has become apparent that they behave as if they have a much larger mass than what could be accounted for by visible objects. There have been many attempts at explaining this invisible mass throughout cosmological history; Could there exist astrophysical objects that are not visible with our current technology? Is there something wrong with our current dynamical models? Or could the mass be accounted for by new, undiscovered particles that do not interact with light? The latter suggestion has become the most promising in the last decades, and the mystical particles are said to constitute so-called dark matter.

There is an abundance of evidence for the existence of DM. One of the most important signs of DM on a galactic scale is the galactic rotation curves, i.e. the graph for the rotational velocity of a galaxy as a function of the distance from the galaxy centre. While Newtonian dynamics predict that the rotation curve decreases inversely proportional to the distance from the galaxy centre outside the optical galaxy disk, observations show that the rotation curves are in fact flat in that area [1]. This implies that there are massive halos outside the optical disk, so-called DM halos.

Other evidence for the existence of DM is the velocity dispersion of astrophysical objects in galaxies. For example, observations of the velocity dispersions of dwarf spheroidal galaxies show larger mass-to-light ratios than expected, which imply the existence of DM halos [2]. Velocity dispersions in galaxy clusters are also strong evidence for DM [3]. Furthermore, quantifications of the dark matter content in the Universe can be found with cosmic microwave background measurements [4].

The leading theory today is that a new type of particle constitutes DM [5]. Many suggestions have been proposed and ruled out as possible DM particles in the last decades, but some remain. Today, the most relevant DM candidates are stable, non-relativistic, electrically neutral and unaffected by strong interactions [5]. One candidate is the axion, a particle that should solve the problem of CP violation in the standard model. Supersymmetric particles, such as the neutralinos, are also widely studied DM candidates. In the last decades, the weakly interacting massive particle (WIMP) has become increasingly interesting, particularly from an experimental perspective [1]. To find the DM particles, they need to be experimentally confirmed. The particles can be discovered either through production in an accelerator experiment, indirect detection of DM annihilation processes, or direct detection of DM particles interacting with detector targets [6].

Since the LHC was finalised in 2008, there have been opportunities for finding traces of DM particles. The DM particle would then be inferred from missing transferred momenta, but no such events have been registered yet [6]. It is still possible to find DM particles and other particles beyond the standard model, for example with the increased performance of the High Luminosity LHC upgrade expected to start operating in 2029 [7].

Indirect detection experiments aim to detect products of DM particle annihilation. The greatest sources for such signals exist in places with large DM densities, such as galactic centres and halos, close cluster galaxies or dwarf galaxies [6]. The product of the self-annihilation process can be pairs of vector bosons, quarks, photons and other combinations of standard model particles. There have been many efforts to detect DM signals through indirect detection, but there have not appeared any clear signals of DM decay. In the cases where signs of DM decay have been possible, it is still unclear if these signals are caused by DM or whether they originate from other astrophysical phenomena [6].

Direct detection experiments are used to detect events where a DM particle interacts directly with the detector target. It has become one of the most promising experimental methods for finding DM particles since many of the DM particles in our galaxy should pass through Earth [1]. Their main purpose has been used to detect WIMPs through nuclear recoil, where a DM particle scatters elastically off detector nucleons [6]. These experiments have not yet led to conclusive results, but the possibility of detecting lighter DM particles that scatter off electrons bound by crystal targets has become an increasingly interesting approach [8].

Today, there are several candidates for DM particles that all obey different theoretical frameworks, and the goal is to test them experimentally. Instead of studying the theories individually, it is convenient to introduce an effective field theory (EFT). In short, EFTs can be described as theories that adequately, but not exactly, describe a physical phenomenon by giving higher importance to certain operators, not unlike a perturbation theory. Most importantly, they can provide model-independent descriptions of scattering processes in direct detection experiments [9]. This introduces a large set of coupling strengths in substitution of a smaller set of model-dependent parameters but has the advantage of e.g. combining all WIMP interactions in one framework.

The power of using EFTs is that when experimental data appears for DM, the

data can be used for estimating the EFT parameters. The parameters can be estimated with Bayes' theorem, which can relate the probability of a set of EFT parameters to the likelihood of the experimental data, i.e. the distribution of data given a set of EFT parameters. In direct detection experiments with crystal detectors, the data is the electronic transition rates in the crystal targets. When such data appears, the distribution of the electronic transition rates for different sets of EFT parameters needs to be found. Obtaining this distribution can be very computationally expensive.

1.2 Aim and outline of the thesis

In this thesis, the electronic transition rates in crystal direct detection targets induced by sub-GeV DM particles are studied. The scattering process is described with an EFT for the scattering amplitude, which introduces a set of EFT coupling strengths. With the DM mass and EFT coupling strengths as input parameters, the electronic transition rates can be computed numerically for different numbers of electron-hole pairs created in the crystal, which was previously performed with a classical Python program. With the purpose of speeding up the computation of the transition rates, an artificial neural network (ANN) was developed. By training the ANN once, a model is created that can repeatedly be used to quickly predict transition rates for different sets of input parameters. Furthermore, the alternative of using graphics processing units (GPUs) to accelerate the computations was investigated.

First, the general electronic transition rate for bound electrons is derived in Chapter 2. The resulting scattering amplitude is then specified for the case of electrons bound in crystals. The chapter concludes with an expression for the transition rate that is used in numerical computations.

Based on the theory of Chapter 2, the transition rates were previously computed in a classical Python program. The intention was for the Python program to generate a training set for the ANN, which in turn is used to speed up predictions of transition rates. However, the Python program itself can be made faster by using C++ and GPU parallelisation. This procedure is outlined in Chapter 3. In Chapter 4, the fundamentals of ANNs are reviewed. Then the theory is applied in an implementation of an ANN for computing the electronic transition rates derived in Chapter 2. The layout of the ANN developed in this thesis is then presented and discussed.

The results of the ANN are presented in Chapter 5, specifically the accuracy and speed. The speed results for the parallelisation of the data generation scripts are also presented and compared to the ANN. The thesis is concluded with Chapter 6, where the results are discussed. Furthermore, improvements and possible extensions of this thesis are suggested.

2

Theoretical description of electronic transition rates

To be able to accelerate the interpretation of direct detection experiments, the governing theory for how DM particles interact with detectors needs to be reviewed. Since this thesis concerns crystal detectors, the central process is DM-electron interactions. More specifically, the rate of electronic transitions in the detector crystals that are induced by DM scattering is studied.

The electronic transition rates are first derived on a general level and are then specified for the case of crystals. The chapter concludes with a summary of the specific observable that is used for numerical calculations, namely the electronic transition rates as a function of the number of electron-hole pairs created in the crystal. Natural units, i.e. $c = \hbar = 1$, are used throughout the whole thesis.

2.1 The general transition rate

We begin by studying the general electronic transition rate for DM-induced excitations of bound electrons. Then it can easily be applied to the case of electrons bound by crystals in Section 2.2. This section largely follows Reference [10].

First, the scattering matrix of the process is derived. Then we proceed by finding the transition rate and conclude the section with a concise expression for the scattering amplitude involving the form factors.

2.1.1 The scattering matrix

The scattering matrix, or S-matrix, describes the transition from an initial state $|i\rangle$ to a final state $|f\rangle$ in a scattering process [11]. Since the electronic transition rate is closely related to the S-matrix, this section is devoted to finding the general S-matrix element for DM scattering against bound electrons. See Figure 2.1 for an example. This process is inelastic and corresponds to a DM particle exciting an electron bound by e.g. a nucleus or a crystal. For the inelastic case, we can define a total state for the DM particle and the electron by a three-dimensional momentum for the DM particle, $\mathbf{p}$, and an electron state, $|\mathbf{e}\rangle$. Following the notation used in Reference [10], we define the initial state $|i\rangle \equiv |\mathbf{e}_1, \mathbf{p}\rangle = |\mathbf{e}_1\rangle \otimes |\mathbf{p}\rangle$ which transitions into the final state $|f\rangle \equiv |\mathbf{e}_2, \mathbf{p}'\rangle$. The electronic states are the energy eigenstates of the electron energy, with eigenenergies E_1 and E_2, respectively. As in Reference

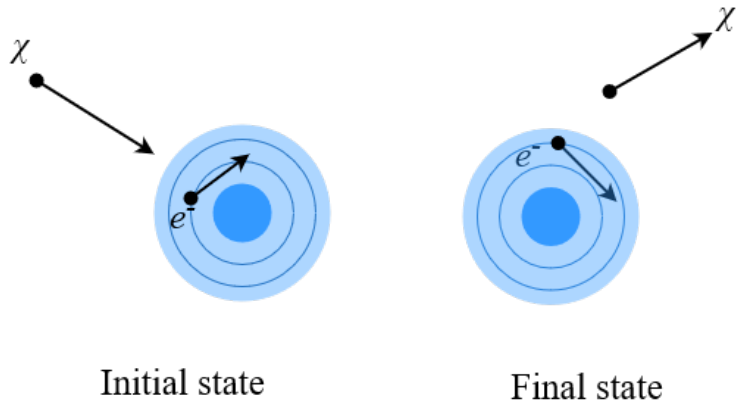


Figure 2.1: An illustration of a dark matter particle, χ, scattering against an electron bound by a nucleus. In the scattering process, the electron transitions into a higher energy level in the atom.

[12], we normalise in a non-relativistic way such that

$$\langle \mathbf{p}|\mathbf{p}\rangle = (2\pi)^3\delta^{(3)}(0) = \int \mathrm{d}^3x \equiv V \tag{2.1}$$

where V is a divergent volume factor that will cancel for all physical observables. Similarly, $\langle \mathbf{e}_1|\mathbf{e}_1\rangle = \langle \mathbf{e}_2|\mathbf{e}_2\rangle = V$ while $\langle \mathbf{e}_1|\mathbf{e}_2\rangle = \langle \mathbf{p}|\mathbf{p}'\rangle = 0$. The dimension of V is energy^{-3}.

Before writing down the general S-matrix element, we develop the Hamiltonian framework for our time-dependent, interacting DM-electron system. We use the interaction picture of quantum mechanics, which is based on the Schrödinger equation

$$i\frac{\mathrm{d}}{\mathrm{d}t}\left|\Phi(t)\right\rangle = H_I(t)\left|\Phi(t)\right\rangle, \tag{2.2}$$

where $|\Phi(t)\rangle$ is our DM-electron state that transitions from $|i\rangle$ to $|f\rangle$ with time, and $H_I(t)$ is the interaction Hamiltonian in the interaction picture [13]. The interaction picture Hamiltonian is non-zero during the finite transition time for our state, but zero in the initial and final states. It is related to the interaction Hamiltonian in the Schrödinger picture, H_S^I, via

$$H_I(t) = e^{iH_S^0 t}H_S^I e^{-iH_S^0 t}, \tag{2.3}$$

where H_S^0 is the interaction-free Hamiltonian in the Schrödinger picture [13, Eqs. (6.12-6.13)]. Since we assume that the transition is adiabatic, i.e. that there is only interaction during the transition between $|i\rangle$ and $|f\rangle$ but no interaction when the system is in $|i\rangle$ or $|f\rangle$, the initial and final states must be eigenstates of H_S^0 [10]. The Hamiltonian, H, in terms of the Hamiltonian density, $\mathscr{H}$, is

$$H = \int \mathrm{d}^3x\mathscr{H}, \tag{2.4}$$

and since H_S^0 is space-independent we have

$$\int \mathrm{d}^3x\mathscr{H}_I(t) = \int \mathrm{d}^3x e^{iH_S^0 t}\mathscr{H}_S^I e^{-iH_S^0 t} \tag{2.5}$$

The transition rate is related to the S-matrix element $\langle f|\, S\, |i\rangle$, as will be described further in Section 2.1.2. The general S-matrix can be expressed with the Dyson expansion [13, (6.23)],

$$S = \sum_{n=0}^{\infty} \frac{(-i)^n}{n!} \int \ldots \int \mathrm{d}^4x_1 \mathrm{d}^4x_2 \ldots \mathrm{d}x_n^4 \mathrm{T}\left\{\mathscr{H}_I(x_1)\mathscr{H}_I(x_2)\ldots\mathscr{H}_I(x_n)\right\}. \tag{2.6}$$

It includes the time-ordering operator, T, which orders the Hamiltonian densities such that the latest operator is placed to the left [11]. The first term in the expansion is just $S^{(0)} = 1$, which implies that the first term in the S-matrix is $\langle f|i\rangle \propto \delta_{fi}$. This corresponds to the case of no transition and is not interesting. To derive the transition rate, we use the first non-trivial S-matrix element in the expansion,

$$S_{fi} \equiv \langle f|\, S^{(1)}\, |i\rangle = -i\,\langle f| \int \mathrm{d}^4x \mathscr{H}_I\, |i\rangle = -i \int \mathrm{d}^4x\, \langle f|\, \mathscr{H}_I\, |i\rangle\,. \tag{2.7}$$

Since the space-time integral over the Hamiltonian density is dimensionless and the overlap of the two-particle states contribute with V^2, the dimension of S_{fi} is energy^{-6}.

Now, by inserting the momentum and electronic state expressions for the initial and final state, as well as using (2.5), we get

$$S_{fi} = -i \int \mathrm{d}^4x\, \langle \mathbf{e}_2, \mathbf{p}'|\, e^{iH_S^0 t} \mathscr{H}_S^I e^{-iH_S^0 t}\, |\mathbf{e}_1, \mathbf{p}\rangle\,. \tag{2.8}$$

Using the fact that $|i\rangle$ and $|f\rangle$ are eigenstates of H_S^0 with eigenenergies E_i and E_f, we get

$$\begin{aligned} S_{fi} &= -i \int \mathrm{d}^4x\, \langle \mathbf{e}_2, \mathbf{p}'|\, \mathscr{H}_S^I\, |\mathbf{e}_1, \mathbf{p}\rangle\, e^{i(E_f - E_i)t} \\ &= -i \int \mathrm{d}^3x\, \langle \mathbf{e}_2, \mathbf{p}'|\, \mathscr{H}_S^I\, |\mathbf{e}_1, \mathbf{p}\rangle \int t e^{i(E_f - E_i)t} \\ &= -i2\pi\delta(E_f - E_i) \int \mathrm{d}^3x\, \langle \mathbf{e}_2, \mathbf{p}'|\, \mathscr{H}_S^I\, |\mathbf{e}_1, \mathbf{p}\rangle\,. \end{aligned} \tag{2.9}$$

In the last step, we inserted the useful property

$$\int \mathrm{d}t e^{ixt} = 2\pi\delta(x), \tag{2.10}$$

which follows from the definition of a Fourier transform. Furthermore, by inserting the completeness relation

$$\int \frac{\mathrm{d}^3\ell}{(2\pi)^3}\, |\boldsymbol{\ell}\rangle\, \langle\boldsymbol{\ell}| = 1 \tag{2.11}$$

twice, where $|\boldsymbol{\ell}\rangle$ are chosen as eigenstates of the Hamiltonian for an unbound electron, we find

$$\begin{aligned} S_{fi} &= -i2\pi\delta(E_f - E_i) \int \mathrm{d}^3x \int \frac{\mathrm{d}^3\ell}{(2\pi)^3} \int \frac{\mathrm{d}^3\ell'}{(2\pi)^3}\, \langle \mathbf{e}_2|\boldsymbol{\ell}'\rangle\, \langle \boldsymbol{\ell}', \mathbf{p}'|\, \mathscr{H}_S^I\, |\mathbf{p}, \boldsymbol{\ell}\rangle\, \langle\boldsymbol{\ell}|\mathbf{e}_1\rangle \\ &= -i2\pi\delta(E_f - E_i) \int \frac{\mathrm{d}^3\ell}{(2\pi)^3} \int \frac{\mathrm{d}^3\ell'}{(2\pi)^3}\, \langle \mathbf{e}_2|\boldsymbol{\ell}'\rangle\, \langle\boldsymbol{\ell}|\mathbf{e}_1\rangle \int \mathrm{d}^3x\, \langle \boldsymbol{\ell}', \mathbf{p}'|\, \mathscr{H}_S^I\, |\mathbf{p}, \boldsymbol{\ell}\rangle\,. \end{aligned} \tag{2.12}$$

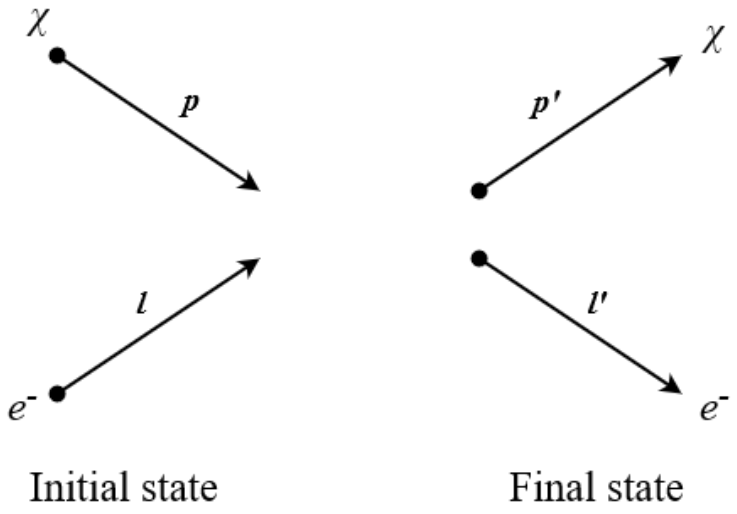


Figure 2.2: An illustration of a dark matter particle, χ, scattering against a free electron. In contrast to the case of a bound electron, the electron state can now be fully expressed with its momentum.

To be able to derive a transition rate from (2.12), we need to evaluate the $\mathscr{H}_S^I$ matrix element. As is soon demonstrated, it is beneficial to look at the elastic scattering of a DM particle against a free electron; see Figure 2.2. In this case, we can denote the initial and final electron states with the electron momentum while keeping the notation for the DM particle, such that $|i\rangle \equiv |\boldsymbol{\ell}, \mathbf{p}\rangle$ and $|f\rangle \equiv |\boldsymbol{\ell}', \mathbf{p}'\rangle$. Their eigenenergies with respect to H_S^0 are $\widetilde{E}_f \equiv E_{\ell'} + E_{\mathbf{p}'}$ and $\widetilde{E}_i \equiv E_\ell + E_{\mathbf{p}}$ [10]. With these energies, the first order free S-matrix element in the Dyson expansion can be found analogously to (2.9),

$$S_{fi}^{\text{free}} = -i2\pi\delta(\widetilde{E}_f - \widetilde{E}_i) \int \mathrm{d}^3x \, \langle \boldsymbol{\ell}', \mathbf{p}' | \, \mathscr{H}_S^I \, | \boldsymbol{\ell}, \mathbf{p} \rangle \, . \tag{2.13}$$

In this expression, we recognise the $\mathscr{H}_S^I$ matrix element from (2.12). To be able to substitute this element we use an expression for the non-trivial free S-matrix element from Reference [11, (4.73)],

$$\begin{aligned} S_{fi}^{\text{free}} &= i(2\pi)^4 \delta^{(4)}(p' + \ell' - p - \ell) \frac{\mathcal{M}(\boldsymbol{\ell}, \mathbf{p}, \boldsymbol{\ell}', \mathbf{p}')}{\sqrt{2E_{\mathbf{p}'} 2E_{\ell'} 2E_{\mathbf{p}} 2E_\ell}} \\ &= i(2\pi)^4 \delta(\widetilde{E}_f - \widetilde{E}_i) \delta^{(3)}(\mathbf{p}' + \boldsymbol{\ell}' - \mathbf{p} - \boldsymbol{\ell}) \frac{\mathcal{M}(\boldsymbol{\ell}, \mathbf{p}, \boldsymbol{\ell}', \mathbf{p}')}{\sqrt{2E_{\mathbf{p}'} 2E_{\ell'} 2E_{\mathbf{p}} 2E_\ell}}. \end{aligned} \tag{2.14}$$

This expression contains the scattering amplitude, $\mathcal{M}$, for the DM-scattering by a free electron. In the second step we used that the 4-momenta p', ℓ', p and ℓ are on-shell, which means $p'^0 = E_{\mathbf{p}'}$, $\ell'^0 = E_{\ell'}$, $p^0 = E_{\mathbf{p}}$ and $\ell^0 = E_\ell$. The factor $1/\sqrt{2E_{\mathbf{p}'} 2E_{\ell'} 2E_{\mathbf{p}} 2E_\ell}$ is not in Reference [11], since they use a relativistic normalisation that corresponds to $\langle \mathbf{p} | \mathbf{p} \rangle = 2E_{\mathbf{p}}(2\pi)^3 \delta^{(3)}(0)$ (compare [11, (2.36)] to our (2.1)). We correct with this factor because $\mathcal{M}$ is assumed to be calculated using Feynman rules with relativistic normalisation.

By comparing (2.13) to (2.14), we see that

$$\begin{aligned}\int \mathrm{d}^3x \, \langle \boldsymbol{\ell}', \mathbf{p}' | \, \mathscr{H}_S^I \, | \boldsymbol{\ell}, \mathbf{p} \rangle = -(2\pi)^3 \delta^{(3)}(\mathbf{p}' + \boldsymbol{\ell}' - \mathbf{p} - \boldsymbol{\ell}) \\ \times \frac{\mathcal{M}(\boldsymbol{\ell}, \mathbf{p}, \boldsymbol{\ell}', \mathbf{p}')}{\sqrt{2E_{\mathbf{p}'} 2E_{\ell'} 2E_{\mathbf{p}} 2E_{\boldsymbol{\ell}}}}.\end{aligned} \tag{2.15}$$

Finally, this can be inserted into (2.12) so that

$$\begin{aligned} S_{fi} &= i2\pi\delta(E_f - E_i) \int \frac{\mathrm{d}^3\ell}{(2\pi)^3} \int \frac{\mathrm{d}^3\ell'}{(2\pi)^3} \langle \mathbf{e}_2 | \boldsymbol{\ell}' \rangle \langle \boldsymbol{\ell} | \mathbf{e}_1 \rangle \\ &\quad \times \delta^{(3)}(\mathbf{p}' + \boldsymbol{\ell}' - \mathbf{p} - \boldsymbol{\ell}) \frac{\mathcal{M}(\boldsymbol{\ell}, \mathbf{p}, \boldsymbol{\ell}', \mathbf{p}')}{\sqrt{2E_{\mathbf{p}'} 2E_{\ell'} 2E_{\mathbf{p}} 2E_{\boldsymbol{\ell}}}} \\ &= i2\pi\delta(E_f - E_i) \int \frac{\mathrm{d}^3\ell}{(2\pi)^3} \langle \mathbf{e}_2 | \boldsymbol{\ell} + \mathbf{q} \rangle \langle \boldsymbol{\ell} | \mathbf{e}_1 \rangle \\ &\quad \times \frac{\mathcal{M}(\boldsymbol{\ell}, \mathbf{p}, \boldsymbol{\ell} + \mathbf{q}, \mathbf{p} - \mathbf{q})}{\sqrt{2E_{\mathbf{p}'} 2E_{\ell'} 2E_{\mathbf{p}} 2E_{\boldsymbol{\ell}}}}, \end{aligned} \tag{2.16}$$

where we introduced the transferred momentum in the scattering process, $\mathbf{q} \equiv \mathbf{p} - \mathbf{p}'$. The integral over ℓ' was resolved by the three-dimensional Dirac delta, which imposed $\boldsymbol{\ell}' = \boldsymbol{\ell} + \mathbf{q}$.

We can introduce wave functions for the electronic states,

$$\begin{aligned} \psi_1(\boldsymbol{\ell}) &\propto \langle \boldsymbol{\ell} | \mathbf{e}_1 \rangle \,, \\ \psi_2(\boldsymbol{\ell} + \mathbf{q}) &\propto \langle \boldsymbol{\ell} + \mathbf{q} | \mathbf{e}_2 \rangle \,, \end{aligned} \tag{2.17}$$

which need to preserve probability [14]. Using Eqs. (2.1) and (2.11), we have

$$| \langle \boldsymbol{\ell} | \mathbf{e}_1 \rangle |^2 = \int \frac{\mathrm{d}^3\ell}{(2\pi)^3} \langle \mathbf{e}_1 | \boldsymbol{\ell} \rangle \langle \boldsymbol{\ell} | \mathbf{e}_1 \rangle = \langle \mathbf{e}_1 | \mathbf{e}_1 \rangle = V, \tag{2.18}$$

and similarly $| \langle \boldsymbol{\ell} + \mathbf{q} | \mathbf{e}_2 \rangle |^2 = V$, so we must have

$$\begin{aligned} \psi_1(\boldsymbol{\ell}) &= \frac{1}{\sqrt{V}} \langle \boldsymbol{\ell} | \mathbf{e}_1 \rangle \,, \\ \psi_2(\boldsymbol{\ell} + \mathbf{q}) &= \frac{1}{\sqrt{V}} \langle \boldsymbol{\ell} + \mathbf{q} | \mathbf{e}_2 \rangle \,. \end{aligned} \tag{2.19}$$

Let us denote the electron mass by m_e and the DM particle mass by m_χ. By exchanging the brackets in (2.16) with these wave functions in the non-relativistic limit, where $2E_{\mathbf{p}'} 2E_{\ell'} 2E_{\mathbf{p}} 2E_{\boldsymbol{\ell}} = 16 m_\chi^2 m_e^2$, we get

$$\begin{aligned} S_{fi} = i2\pi\delta(E_f - E_i) \frac{V}{4 m_\chi m_e} \int \frac{\mathrm{d}^3\ell}{(2\pi)^3} \psi_2^*(\boldsymbol{\ell} + \mathbf{q}) \psi_1(\boldsymbol{\ell}) \\ \times \mathcal{M}(\boldsymbol{\ell}, \mathbf{p}, \boldsymbol{\ell} + \mathbf{q}, \mathbf{p} - \mathbf{q}). \end{aligned} \tag{2.20}$$

With this, we have an expression for the general S-matrix element in terms of the scattering amplitude and the wave functions for bound electrons bound. This will now be used to derive the electronic transition rates and will be specified for electrons bound by crystals in Section 2.2.

2.1.2 The general transition rate

The probability for the transition $|\mathbf{e}_1, \mathbf{p}\rangle \to |\mathbf{e}_2, \mathbf{p}'\rangle$ between two definitive states is

$$\mathrm{P} = \frac{|S_{fi}|^2}{V^4}. \tag{2.21}$$

The factor $1/V^4$ is needed to normalise the probability since we have discretised momenta [8]. For S_{fi}, we use (2.20). Now we want to calculate the probability of a transition from a specific state with DM momentum $\mathbf{p}$ to a state with final DM momentum in the interval $(\mathbf{p}', \mathbf{p}' + \mathrm{d}\mathbf{p}')$. The number of such final states is [13]

$$N = \frac{V\mathrm{d}^3p'}{(2\pi)^3} = \frac{V\mathrm{d}^3q}{(2\pi)^3}, \tag{2.22}$$

which leads to the total transition probability,

$$\mathscr{P}(\mathbf{p}) = \mathrm{P} \cdot N = \frac{|S_{fi}|^2}{V^4}\frac{V\mathrm{d}^3q}{(2\pi)^3}. \tag{2.23}$$

To be able to square the energy Dirac delta in (2.20), we define a divergent factor

$$T \equiv \int \mathrm{d}t = \int_{-T/2}^{T/2} \mathrm{d}t, \tag{2.24}$$

similarly to how we defined V, and use Reference [13, Eqs. (8.4-8.5)] for large T to show that

$$[2\pi\delta(E_f - E_i)]^2 = T(2\pi)\delta(E_f - E_i). \tag{2.25}$$

Now the expression for S_{fi} in (2.20) can be inserted into (2.23) so that

$$\begin{aligned}\mathscr{P}(\mathbf{p}) = 2\pi\delta(E_f - E_i)&\frac{1}{16m_\chi^2 m_e^2}\frac{T\mathrm{d}^3q}{(2\pi)^3V} \\ &\times \left|\int \frac{\mathrm{d}^3\ell}{(2\pi)^3}\psi_2^*(\boldsymbol{\ell} + \mathbf{q})\mathcal{M}(\boldsymbol{\ell}, \mathbf{p}, \mathbf{q})\psi_1(\boldsymbol{\ell})\right|^2.\end{aligned} \tag{2.26}$$

The transition rate is defined as the transition probability per unit time, i.e. $\mathscr{P}(\mathbf{p})/T$, so the divergent time factor in the transition probability is cancelled. The divergent volume V is defined such that it contains exactly one DM particle, which means that the unit density for DM particles is $1/V$ [13]. Thus, we can define the transition rate per unit DM density as $V\mathscr{P}(\mathbf{p})/T$. To get back to a transition rate, we now multiply this quantity by the local number density for DM particles, $n_\chi = 0.4\ \mathrm{GeV/cm^3}/m_\chi$ [10]. Finally, we get the total transition rate by allowing all combinations of incoming and outgoing momenta. This requires integrating the transition rate over the incoming DM particle velocity distribution, $f_\chi(\mathbf{v})$, and the transition momentum $\mathbf{q}$ [10],

$$\mathscr{R}_{1\to2} = 2\pi\delta(E_f - E_i)\frac{n_\chi}{16m_\chi^2 m_e^2}\int \frac{\mathrm{d}^3q}{(2\pi)^3}\int d^3v f_\chi(v)\overline{|\mathcal{M}_{1\to2}|^2}, \tag{2.27}$$

where

$$\overline{|\mathcal{M}_{1\to 2}|^2} \equiv \overline{\left| \int \frac{\mathrm{d}^3\ell}{(2\pi)^3} \psi_2^*(\boldsymbol{\ell}+\mathbf{q})\mathcal{M}(\boldsymbol{\ell},\mathbf{p},\mathbf{q})\psi_1(\boldsymbol{\ell}) \right|^2} \tag{2.28}$$

and the bar over the squared transition amplitude corresponds to averaging over incoming spin states and summing over outgoing spin states [11]. Following Reference [10], we use the velocity distribution

$$f_\chi(\mathbf{v}) = \frac{1}{N_{\mathrm{esc}}\pi^{3/2}v_0^3} \exp\left[-\frac{(\mathbf{v}+\mathbf{v}_\oplus)^2}{v_0^2}\right] \Theta(v_{\mathrm{esc}} - |\mathbf{v}+\mathbf{v}_\oplus|) \tag{2.29}$$

with

$$N_{\mathrm{esc}} \equiv \mathrm{erf}(v_{\mathrm{esc}}/v_0) - 2(v_{\mathrm{esc}}/v_0)\exp\left(-v_{\mathrm{esc}}^2/v_0^2\right)/\sqrt{\pi} \tag{2.30}$$

and $v_0 = 220$ km/s, $v_{\mathrm{esc}} = 544$ km/s, $v_\oplus = 244$ km/s.

Equation (2.27) is the transition rate for the case of DM-electron scattering for bound electrons and will be used in Section 2.2. Before proceeding to the case of electrons bound by crystals, however, the scattering amplitude can be simplified.

2.1.3 Expansion in the electron momentum-to-mass ratio

The total transition rate is given by (2.27), so the rest of this chapter is dedicated to rewriting (2.28). Here, we make a general expansion that can be used in the case of electrons bound by crystals.

Up until this point we have been working with the incoming and outgoing DM- and electron momenta $\mathbf{p}'$, $\boldsymbol{\ell}'$, $\mathbf{p}$ and $\boldsymbol{\ell}$. However, since DM-free electron scattering is characterised by momentum conservation and Galilean invariance, only two of these momenta are independent [8]. We have introduced the transferred momentum $\mathbf{q} = \mathbf{p} - \mathbf{p}'$, but we can also choose an independent momentum proportional to [9]

$$\begin{aligned}
\mathbf{v}_{\mathrm{el}}^{\perp} &= \frac{\mathbf{p}+\mathbf{p}'}{2m_\chi} - \frac{\boldsymbol{\ell}+\boldsymbol{\ell}'}{2m_e} \\
&= \frac{\mathbf{p}}{m_\chi} - \frac{\mathbf{p}-\mathbf{p}'}{2m_\chi} - \frac{\boldsymbol{\ell}+\boldsymbol{\ell}+\mathbf{q}}{2m_e} \\
&= \mathbf{v}_{\mathrm{rel,in}} - \frac{\mathbf{q}}{2}\left(\frac{1}{m_\chi} + \frac{1}{m_e}\right) \\
&= \mathbf{v}_{\mathrm{rel,in}} - \frac{\mathbf{q}}{2\mu_{\chi e}},
\end{aligned} \tag{2.31}$$

where we used $\boldsymbol{\ell}' = \boldsymbol{\ell} + \mathbf{q}$ [10]. We introduced the reduced mass of the DM particle and electron, $\mu_{\chi e}$, and the relative incoming velocity

$$\mathbf{v}_{\mathrm{rel,in}} = \frac{\mathbf{p}}{m_\chi} - \frac{\boldsymbol{\ell}}{m_e}. \tag{2.32}$$

The relative outgoing velocity is

$$\begin{aligned}\mathbf{v}_{\text{rel,out}} &= \frac{\mathbf{p}'}{m_\chi} - \frac{\boldsymbol{\ell}'}{m_e} \\ &= \frac{\mathbf{p}}{m_\chi} - \frac{\mathbf{p}-\mathbf{p}'}{m_\chi} - \frac{\boldsymbol{\ell}+\mathbf{q}}{m_e} \\ &= \mathbf{v}_{\text{rel,in}} - \frac{\mathbf{q}}{\mu_{\chi e}}.\end{aligned} \tag{2.33}$$

This leads to the initial and final kinetic energies $E_i^{\text{CM}} = \mu_{\chi e}\mathbf{v}_{\text{rel,in}}^2/2$ and $E_f^{\text{CM}} = \mu_{\chi e}\mathbf{v}_{\text{rel,out}}^2/2$ in the centre of mass frame. In elastic scattering processes, energy conservation implies

$$\begin{aligned}\mathbf{v}_{\text{rel,in}}^2 &= \mathbf{v}_{\text{rel,out}}^2 \\ &= \left(\mathbf{v}_{\text{rel,in}} - \frac{\mathbf{q}}{\mu_{\chi e}}\right)^2 \\ &= \mathbf{v}_{\text{rel,in}}^2 - \frac{2(\mathbf{v}_{\text{rel,in}}\cdot\mathbf{q})}{\mu_{\chi e}} + \frac{\mathbf{q}^2}{\mu_{\chi e}^2},\end{aligned} \tag{2.34}$$

which leads to

$$\mathbf{v}_{\text{rel,in}}\cdot\mathbf{q} = \frac{\mathbf{q}^2}{2\mu_{\chi e}}. \tag{2.35}$$

Now it is apparent why $\mathbf{v}_{\text{el}}^\perp$ is a good choice of independent momentum; in elastic processes,

$$\mathbf{v}_{\text{el}}^\perp\cdot\mathbf{q} = 0. \tag{2.36}$$

The scattering amplitude $\mathcal{M}$ can now be expressed using only $\mathbf{q}$ and $\mathbf{v}_{\text{el}}^\perp$. In the non-relativistic limit, we can expand $\mathcal{M}$ in the electron momentum-to-mass ratio, $|\boldsymbol{\ell}|/m_e \ll 1$ [8],

$$\mathcal{M}(\mathbf{q},\mathbf{v}_{\text{el}}^\perp) \approx \left[\mathcal{M}(\mathbf{q},\mathbf{v}_{\text{el}}^\perp)\right]_{\boldsymbol{\ell}=0} + \left(\frac{\boldsymbol{\ell}}{m_e}\right)\cdot\left[m_e\nabla_{\boldsymbol{\ell}}\mathcal{M}(\mathbf{q},\mathbf{v}_{\text{el}}^\perp)\right]_{\boldsymbol{\ell}=0}. \tag{2.37}$$

From this point onward, we write $\mathcal{M} \equiv \mathcal{M}(\mathbf{q},\mathbf{v}_{\text{el}}^\perp)$. The scattering amplitudes on the right-hand side of (2.37) will be expressed using an effective theory in Section 2.1.4. In preparation for that, we further simplify the expression here.

To begin with, we can insert (2.37) in (2.28) to get

$$\overline{|\mathcal{M}_{1\to2}|^2} = \overline{\left|\int\frac{\mathrm{d}^3\ell}{(2\pi)^3}\psi_2^*(\boldsymbol{\ell}+\mathbf{q})\left[\mathcal{M}_{\boldsymbol{\ell}=0} + \left(\frac{\boldsymbol{\ell}}{m_e}\right)\cdot(m_e\nabla_{\boldsymbol{\ell}}\mathcal{M})_{\boldsymbol{\ell}=0}\right]\psi_1(\boldsymbol{\ell})\right|^2}, \tag{2.38}$$

which can be rewritten as

$$\begin{aligned}\overline{|\mathcal{M}_{1\to2}|^2} = &\overline{\left|\int \frac{\mathrm{d}^3\ell}{(2\pi)^3}\psi_2^*(\boldsymbol{\ell}+\mathbf{q})\mathcal{M}_{\boldsymbol{\ell}=0}\psi_1(\boldsymbol{\ell})\right|^2} \\ &+ \overline{2\,\mathrm{Re}\left\{\left[\int \frac{\mathrm{d}^3\ell}{(2\pi)^3}\psi_2^*(\boldsymbol{\ell}+\mathbf{q})\mathcal{M}_{\boldsymbol{\ell}=0}\psi_1(\boldsymbol{\ell})\right]\right.} \\ &\qquad \left.\overline{\times\left[\int \frac{\mathrm{d}^3\ell}{(2\pi)^3}\psi_2(\boldsymbol{\ell}+\mathbf{q})\left(\frac{\boldsymbol{\ell}}{m_e}\right)\cdot\left[m_e\nabla_{\boldsymbol{\ell}}\mathcal{M}^*\right]_{\boldsymbol{\ell}=0}\psi_1^*(\boldsymbol{\ell})\right]}\right\} \\ &+ \overline{\left|\int \frac{\mathrm{d}^3\ell}{(2\pi)^3}\psi_2^*(\boldsymbol{\ell}+\mathbf{q})\left(\frac{\boldsymbol{\ell}}{m_e}\right)\cdot\left[m_e\nabla_{\boldsymbol{\ell}}\mathcal{M}\right]_{\boldsymbol{\ell}=0}\psi_1(\boldsymbol{\ell})\right|^2}.\end{aligned} \tag{2.39}$$

by expanding the square of the two terms. Since the scattering amplitudes are evaluated at $\boldsymbol{\ell}=0$, they can be moved out of the integrals so that

$$\begin{aligned}\overline{|\mathcal{M}_{1\to2}|^2} = &\overline{|\mathcal{M}|^2_{\boldsymbol{\ell}=0}}\overline{\left|\int \frac{\mathrm{d}^3\ell}{(2\pi)^3}\psi_2^*(\boldsymbol{\ell}+\mathbf{q})\psi_1(\boldsymbol{\ell})\right|^2} \\ &+ 2m_e\overline{\mathrm{Re}\left\{\mathcal{M}_{\boldsymbol{\ell}=0}\left[\int \frac{\mathrm{d}^3\ell}{(2\pi)^3}\psi_2^*(\boldsymbol{\ell}+\mathbf{q})\psi_1(\boldsymbol{\ell})\right]\right.} \\ &\qquad \left.\overline{\times\left[\nabla_{\boldsymbol{\ell}}\mathcal{M}^*\right]_{\boldsymbol{\ell}=0}\cdot\left[\int \frac{\mathrm{d}^3\ell}{(2\pi)^3}\psi_2(\boldsymbol{\ell}+\mathbf{q})\left(\frac{\boldsymbol{\ell}}{m_e}\right)\psi_1^*(\boldsymbol{\ell})\right]}\right\} \\ &+ m_e^2\overline{\left|\left[\nabla_{\boldsymbol{\ell}}\mathcal{M}\right]_{\boldsymbol{\ell}=0}\cdot\int \frac{\mathrm{d}^3\ell}{(2\pi)^3}\psi_2^*(\boldsymbol{\ell}+\mathbf{q})\left(\frac{\boldsymbol{\ell}}{m_e}\right)\psi_1(\boldsymbol{\ell})\right|^2}.\end{aligned} \tag{2.40}$$

This motivates the definitions of the general electron wave function overlap integrals, or form factors [10],

$$f_{1\to2}(\mathbf{q}) \equiv \int \frac{\mathrm{d}^3\ell}{(2\pi)^3}\psi_2^*(\boldsymbol{\ell}+\mathbf{q})\psi_1(\boldsymbol{\ell}) \tag{2.41}$$

and

$$\mathbf{f}_{1\to2}(\mathbf{q}) \equiv \int \frac{\mathrm{d}^3\ell}{(2\pi)^3}\psi_2^*(\boldsymbol{\ell}+\mathbf{q})\left(\frac{\boldsymbol{\ell}}{m_e}\right)\psi_1(\boldsymbol{\ell}). \tag{2.42}$$

By inserting these in (2.40), we get

$$\begin{aligned}\overline{|\mathcal{M}_{1\to2}|^2} = &\overline{|\mathcal{M}|^2_{\boldsymbol{\ell}=0}}\,|f_{1\to2}|^2 + 2m_e\overline{\mathrm{Re}\left[\mathcal{M}_{\boldsymbol{\ell}=0}f_{1\to2}\left[\nabla_{\boldsymbol{\ell}}\mathcal{M}^*\right]_{\boldsymbol{\ell}=0}\cdot\mathbf{f}_{1\to2}^*\right]} \\ &+ m_e^2\overline{|\left[\nabla_{\boldsymbol{\ell}}\mathcal{M}\right]_{\boldsymbol{\ell}=0}\cdot\mathbf{f}_{1\to2}|^2},\end{aligned} \tag{2.43}$$

or even more concisely,

$$\overline{|\mathcal{M}_{1\to2}|^2} = \left\{\overline{|\mathcal{M}|^2}\,|f_{1\to2}|^2 + 2m_e\overline{\mathrm{Re}\left(\mathcal{M}f_{1\to2}\nabla_{\boldsymbol{\ell}}\mathcal{M}^*\cdot(\mathbf{f}_{1\to2})^*\right)} + m_e^2\overline{|\nabla_{\boldsymbol{\ell}}\mathcal{M}\cdot\mathbf{f}_{1\to2}|^2}\right\}_{\boldsymbol{\ell}=0}. \tag{2.44}$$

Expressing the scattering amplitude with the form factors enables a factorisation between the scattering properties, which are contained in $\mathcal{M}$, and the material-specific properties contained in the form factors. It will also make the step from the general transition rate to the crystal transition rate more clear.

2.1.4 Expansion with effective field theory

We have seen that the scattering amplitude can be described with the independent variables $\mathbf{v}_{\text{el}}^{\perp}$ and $\mathbf{q}$. Furthermore, it is characterised by the spin operators $\mathbf{S}_e$ and $\mathbf{S}_\chi$ for the electron and DM particle, assuming the DM particle has intrinsic spin [9]. We already expanded the scattering matrix in the electron momentum-to-mass ratio in Section 2.1.3. To express the scattering amplitudes in the expanded expression, such as the right-hand side of (2.44), we use effective theory [10, 8, 9].

An effective theory can model a physical phenomenon with a large set of parameters in a perturbative manner, in contrast to relying on a specific underlying theory [15]. In the case of DM research, effective theories have the advantage of being able to explore different DM particle candidates by altering the coefficients of the effective theory. It is outside the scope of this thesis to investigate effective theories in detail, therefore the effective theory used in this thesis will just be presented here.

Using the EFT described in References [8, 9] corresponds to writing the scattering amplitude as

$$\mathcal{M}(\mathbf{q}, \mathbf{v}_{\text{el}}^{\perp}) = \sum_i \left(c_i^s + c_i^\ell \frac{q_{\text{ref}}^2}{|\mathbf{q}|^2} \right) \langle \mathcal{O}_i \rangle, \tag{2.45}$$

where $\mathcal{O}_i$ are interaction operators, and c_i^s and c_i^ℓ are short- and long-range interaction coupling strengths. The reference momentum q_{ref} is defined as $q_{\text{ref}} \equiv \alpha m_e$, where α is the fine structure constant. The angled brackets in (2.45) denote the matrix element for an interaction operator acting on the initial and final two-component spinors ξ^λ and $\xi^{\lambda'}$ for the electron state, and ξ^s and $\xi^{s'}$ for the DM particle state, i.e.

$$\langle \mathcal{O}_i \rangle \equiv \xi^{s'} \xi^s \mathcal{O}_i \xi^{\lambda'} \xi^\lambda. \tag{2.46}$$

In this thesis, we follow Reference [10, 8] and use operators $\mathcal{O}_i$ that describe fermionic DM particles up to first order in $\mathbf{v}_{\text{el}}^{\perp}$ and second order in $\mathbf{q}$. These operators are [8, 9]

$$\begin{cases} \mathcal{O}_1 = \mathbf{1}_\chi \mathbf{1}_e, \\ \mathcal{O}_3 = i\mathbf{1}_\chi \mathbf{S}_e \cdot (\mathbf{q} \times \mathbf{v}_{\text{el}}^{\perp}), \\ \mathcal{O}_4 = \mathbf{S}_\chi \cdot \mathbf{S}_e, \\ \mathcal{O}_5 = i\mathbf{S}_\chi \cdot (\mathbf{q} \times \mathbf{v}_{\text{el}}^{\perp}) \mathbf{1}_e, \\ \mathcal{O}_6 = (\mathbf{S}_\chi \cdot \mathbf{q})(\mathbf{S_e} \cdot \mathbf{q}), \\ \mathcal{O}_7 = \mathbf{1}_\chi (\mathbf{S}_e \cdot \mathbf{v}_{\text{el}}^{\perp}), \\ \mathcal{O}_8 = (\mathbf{S}_\chi \cdot \mathbf{v}_{\text{el}}^{\perp}) \mathbf{1}_e, \\ \mathcal{O}_9 = i\mathbf{S}_\chi \cdot (\mathbf{S}_e \times \mathbf{q}), \\ \mathcal{O}_{10} = i\mathbf{1}_\chi (\mathbf{S}_e \cdot \mathbf{q}), \\ \mathcal{O}_{11} = i(\mathbf{S}_\chi \cdot \mathbf{q}) \mathbf{1}_e, \\ \mathcal{O}_{12} = \mathbf{S}_\chi \cdot (\mathbf{S}_e \times \mathbf{v}_{\text{el}}^{\perp}) \\ \mathcal{O}_{13} = i(\mathbf{S}_\chi \cdot \mathbf{v}_{\text{el}}^{\perp})(\mathbf{S}_e \cdot \mathbf{q}) \\ \mathcal{O}_{14} = i(\mathbf{S}_\chi \cdot \mathbf{q})(\mathbf{S}_e \cdot \mathbf{v}_{\text{el}}^{\perp}) \\ \mathcal{O}_{15} = i\mathcal{O}_{11}[(\mathbf{S}_e \times \mathbf{v}_{\text{el}}^{\perp}) \cdot \mathbf{q}]. \end{cases} \tag{2.47}$$

Since $\mathcal{O}_2 = \mathbf{1}_\chi \mathbf{1}_e (\mathbf{v}_{\text{el}}^\perp)^2$ is not linear in $\mathbf{v}_{\text{el}}^\perp$, it is excluded in the set of operators used here. From (2.45) we see that these 14 operators lead to a total of 28 short- and long-range coupling strengths.

Before applying the effective theory to express the scattering amplitude, the transition rate for crystal targets will be derived. It is not until we arrive at a final crystal transition rate that the effective theory will enter, in the form of DM response functions; see Section 2.2.4.

2.2 The electronic transition rate in crystals

The transition rate in (2.27) together with (2.44) is general and can be applied to any target material in a direct detection experiment. In this section, the transition rate is specified for the case of crystal targets. To this end, a formalism for electrons bound to crystals is set up, followed by a specification of the scattering amplitude. Then, the new framework is applied to (2.27). This section largely follows References [10, 8].

2.2.1 Crystal wave function formalism

Specifying the electrons to be bound by crystals corresponds to specifying the form factors, and thus wave functions, in (2.44). These electrons exist in the valence bands of the crystal in the initial state and transition to the conduction bands for the final state. See Figure 2.3. Thus, we define the electronic state by its energy band index, j, and its wave vector, $\mathbf{k}$, in the first Brillouin zone (BZ). Note that these $\mathbf{k}$ are not the electron momenta, but the wave vectors of a Bloch wave. They can also be interpreted as the quantum number of an electronic state [16].

The electronic transition can be described as $j\mathbf{k} \to j'\mathbf{k}'$. The initial band index j spans over all valence bands in the crystal, while the final band index j' spans over all conduction bands. With the normalisation of the wave functions introduced in

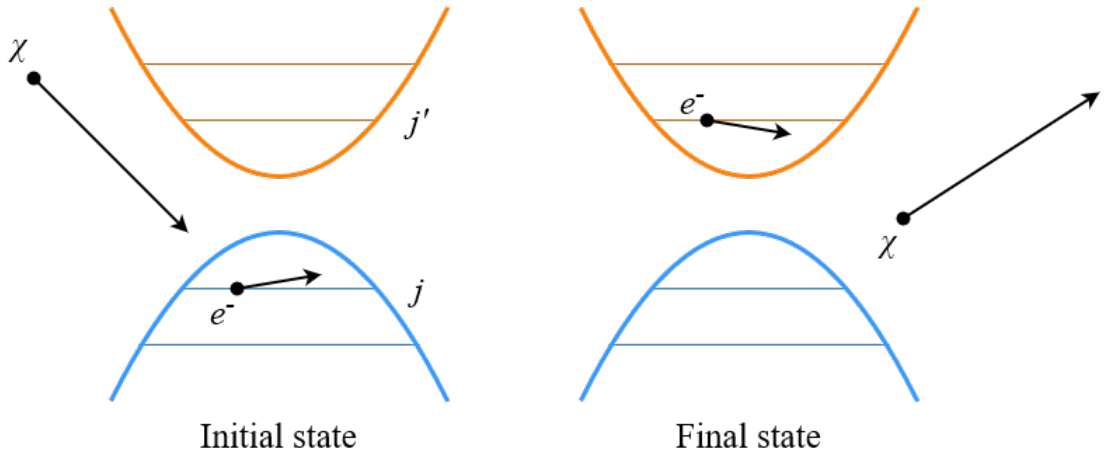


Figure 2.3: An illustration of the case of a dark matter particle, χ scattered against an electron bound by a crystal. In this case, the electron transitions from the valence band j into the conduction band j'.

(2.19), the electronic position space wave function at $\mathbf{x}$ can be expressed as a sum over the reciprocal lattice vectors $\mathbf{G}$,

$$\psi_{j\mathbf{k}}(\mathbf{x}) = \frac{1}{\sqrt{V}} \sum_{\mathbf{G}} u_j(\mathbf{k}+\mathbf{G}) e^{i(\mathbf{k}+\mathbf{G})\cdot\mathbf{x}}, \tag{2.48}$$

and analogously for $\psi_{j'\mathbf{k}'}$ [16]. Preservation of probability for the position space wave function leads to a condition for the coefficients u_j,

$$\begin{aligned} \int_V \mathrm{d}^3x |\psi_{j\mathbf{k}}|^2 &= \int_V \mathrm{d}^3x \left| \frac{1}{\sqrt{V}} \sum_{\mathbf{G}} u_j(\mathbf{k}+\mathbf{G}) e^{i(\mathbf{k}+\mathbf{G})\cdot\mathbf{x}} \right|^2 \\ &= \frac{1}{V} \int_V \mathrm{d}^3x \sum_{\mathbf{G}} |u_j(\mathbf{k}+\mathbf{G})|^2 \\ &= \sum_{\mathbf{G}} |u_j(\mathbf{k}+\mathbf{G})|^2 \\ &\equiv 1, \end{aligned} \tag{2.49}$$

where Parseval's formula was used in the second step [17, (3.22)].

In Section 2.1, V was referred to as a divergent volume; here it becomes the volume of the crystal detector, such that $V = N_{\mathrm{cell}} V_{\mathrm{cell}}$ where V_{cell} is the unit cell volume and N_{cell} is the number of unit cells in the crystal. We have $N_{\mathrm{cell}} = M_{\mathrm{target}}/M_{\mathrm{cell}}$ where M_{target} is the total detector target mass and M_{cell} is the mass of a single unit cell. Both germanium and silicon have face-centred diamond-cubic crystal structures, which means there are eight atoms in each unit cell, i.e. $M_{\mathrm{cell}} = 8m_{\mathrm{Si/Ge}}$. The unit cell crystal structure of silicon and germanium consists of several copies of two base cell components; an empty cell, and a cell with a total of two atoms. In some literature, such as Reference [8], the mass of the non-empty base cells ($2m_{\mathrm{Si/Ge}}$) is referred to as M_{cell}. This is consistent with what is stated here, as long as N_{cell} is changed accordingly.

Let $\mathscr{R}_{j\mathbf{k}\to j'\mathbf{k}'}$ denote the transition rate in (2.27) expressed with the position space wave functions. This is the transition rate for electrons in specific energy bands and quantum states, but the relevant observable in DM direct detection experiments is the full electronic transition rate in crystals [8]. Thus, we need to sum $\mathscr{R}_{j\mathbf{k}\to j'\mathbf{k}'}$ over both the input and output crystal volume and electron wave vectors as well as energy bands and spin states,

$$\mathscr{R}_{\mathrm{crystal}} = 2 \sum_{jj'} \int_{\mathrm{BZ}} \frac{V \mathrm{d}^3k}{(2\pi)^3} \int_{\mathrm{BZ}} \frac{V \mathrm{d}^3k'}{(2\pi)^3} \mathscr{R}_{j\mathbf{k}\to j'\mathbf{k}'}. \tag{2.50}$$

The factor of two is caused by double occupancy in each crystal conductive band since each energy band can contain one spin-up and one spin-down electron.

Since we are using position space wave functions instead in (2.50), we need to express (2.28) in terms of position space wave functions. The momentum and position space wave functions are related with a Fourier transform,

$$\psi_1(\boldsymbol{\ell}) = \int \mathrm{d}^3x \psi_{j\mathbf{k}}(\mathbf{x}) e^{-i\boldsymbol{\ell}\cdot\mathbf{x}}. \tag{2.51}$$

Thus, we can rewrite the scalar form factor in (2.41),

$$\begin{aligned}\int \frac{\mathrm{d}^3\ell}{(2\pi)^3}\psi_2^*(\boldsymbol{\ell}+\mathbf{q})\psi_1(\boldsymbol{\ell}) &= \int \frac{\mathrm{d}^3\ell}{(2\pi)^3}\int \mathrm{d}^3x'\psi^*_{j'\mathbf{k}'}(\mathbf{x}')e^{i(\boldsymbol{\ell}+\mathbf{q})\cdot\mathbf{x}'}\int \mathrm{d}^3x\psi_{j\mathbf{k}}(\mathbf{x})e^{-i\boldsymbol{\ell}\cdot\mathbf{x}} \\ &= \int \mathrm{d}^3x\int \mathrm{d}^3x'\psi^*_{j'\mathbf{k}'}(\mathbf{x}')e^{i\mathbf{q}\cdot\mathbf{x}'}\psi_{j\mathbf{k}}(\mathbf{x})\int \frac{\mathrm{d}^3\ell}{(2\pi)^3}e^{-i\boldsymbol{\ell}\cdot(\mathbf{x}-\mathbf{x}')} \\ &= \int \mathrm{d}^3x\int \mathrm{d}^3x'\psi^*_{j'\mathbf{k}'}(\mathbf{x}')e^{i\mathbf{q}\cdot\mathbf{x}'}\psi_{j\mathbf{k}}(\mathbf{x})\delta^{(3)}(\mathbf{x}-\mathbf{x}') \\ &= \int \mathrm{d}^3x\psi^*_{j'\mathbf{k}'}(\mathbf{x})e^{i\mathbf{q}\cdot\mathbf{x}}\psi_{j\mathbf{k}}(\mathbf{x}),\end{aligned} \tag{2.52}$$

where we used (2.10) in the third step and resolved the Dirac delta in the last step. As for rewriting the vectorial form factor in (2.42), we have the following Fourier transform property [18, F11],

$$\begin{aligned}\mathcal{F}\left\{\left(\frac{\boldsymbol{\ell}}{m_e}\right)\psi_1(\boldsymbol{\ell})\right\}(\mathbf{x}) &= \frac{i\nabla_{\mathbf{x}}}{m_e}\mathcal{F}\{\psi_1(\boldsymbol{\ell})\}(\mathbf{x}) \\ &= \frac{i\nabla_{\mathbf{x}}}{m_e}\psi_{j\mathbf{k}}(\mathbf{x})\end{aligned} \tag{2.53}$$

or

$$\left(\frac{\boldsymbol{\ell}}{m_e}\right)\psi_1(\boldsymbol{\ell}) = \int \mathrm{d}^3x\frac{i\nabla_{\mathbf{x}}}{m_e}\psi_{j\mathbf{k}}(\mathbf{x})e^{-i\boldsymbol{\ell}\cdot\mathbf{x}}. \tag{2.54}$$

This implies that (2.42) can be written as

$$\int \frac{\mathrm{d}^3\ell}{(2\pi)^3}\psi_2^*(\boldsymbol{\ell}+\mathbf{q})\left(\frac{\boldsymbol{\ell}}{m_e}\right)\psi_1(\boldsymbol{\ell}) = \int \mathrm{d}^3x\psi_{j'\mathbf{k}'}(\mathbf{x})e^{i\mathbf{q}\cdot\mathbf{x}}\frac{i\nabla_{\mathbf{x}}}{m_e}\psi_{j\mathbf{k}}(\mathbf{x}). \tag{2.55}$$

Finally, we see that by defining the scalar

$$f_{j\mathbf{k}\to j'\mathbf{k}'}(\mathbf{q}) \equiv \int \mathrm{d}^3x\psi^*_{j'\mathbf{k}'}(\mathbf{x})e^{i\mathbf{q}\cdot\mathbf{x}}\psi_{j\mathbf{k}}(\mathbf{x}) \tag{2.56}$$

and the 3-vector

$$\mathbf{f}_{j\mathbf{k}\to j'\mathbf{k}'}(\mathbf{q}) \equiv \int \mathrm{d}^3x\psi_{j'\mathbf{k}'}(\mathbf{x})e^{i\mathbf{q}\cdot\mathbf{x}}\frac{i\nabla_{\mathbf{x}}}{m_e}\psi_{j\mathbf{k}}(\mathbf{x}), \tag{2.57}$$

we can write (2.44) in terms of the electronic wave functions for crystals,

$$\begin{aligned}\overline{|\mathcal{M}_{j\mathbf{k}\to j'\mathbf{k}'}|^2} = \Bigg\{&\overline{|\mathcal{M}|^2}\,|f_{j\mathbf{k}\to j'\mathbf{k}'}|^2 + 2m_e\overline{\mathrm{Re}\left[\mathcal{M}f_{j\mathbf{k}\to j'\mathbf{k}'}\nabla_{\boldsymbol{\ell}}\mathcal{M}^*\cdot(\mathbf{f}_{j\mathbf{k}\to j'\mathbf{k}'})^*\right]} \\ &+ m_e^2\overline{|\nabla_{\boldsymbol{\ell}}\mathcal{M}\cdot\mathbf{f}_{j\mathbf{k}\to j'\mathbf{k}'}|^2}\Bigg\}_{\boldsymbol{\ell}=0}.\end{aligned} \tag{2.58}$$

The scattering amplitude is now expressed in terms of the position space electronic wave functions for crystals. However, before computing the total transition rate, we can simplify the scattering amplitude.

2.2.2 Rewriting the scattering amplitude

To find the total transition rate corresponding to (2.58), it turns out that it is beneficial to extract a three-dimensional Dirac delta. We start by writing the form factors more explicitly using the expressions for the crystal wave functions from (2.48). Then (2.56) becomes

$$\begin{aligned} f_{j\mathbf{k}\to j'\mathbf{k}'}(\mathbf{q}) &= \frac{1}{V}\int \mathrm{d}^3x \left[\sum_{\mathbf{G}'} u^*_{j'}(\mathbf{k}'+\mathbf{G}')e^{-i(\mathbf{k}'+\mathbf{G}')\cdot\mathbf{x}}\right] e^{i\mathbf{q}\cdot\mathbf{x}} \left[\sum_{\mathbf{G}} u_j(\mathbf{k}+\mathbf{G})e^{i(\mathbf{k}+\mathbf{G})\cdot\mathbf{x}}\right] \\ &= \frac{1}{V}\sum_{\mathbf{G}\mathbf{G}'} u^*_{j'}(\mathbf{k}'+\mathbf{G}')u_j(\mathbf{k}+\mathbf{G})\int \mathrm{d}^3x e^{i(\mathbf{k}-\mathbf{k}'+\mathbf{G}-\mathbf{G}'+\mathbf{q})\cdot\mathbf{x}} \\ &= \frac{(2\pi)^3}{V}\sum_{\mathbf{G}\mathbf{G}'} u^*_{j'}(\mathbf{k}'+\mathbf{G}')u_j(\mathbf{k}+\mathbf{G})\delta^{(3)}(\mathbf{k}-\mathbf{k}'+\mathbf{G}-\mathbf{G}'+\mathbf{q}), \end{aligned} \tag{2.59}$$

where (2.10) was used in the last step. Similarly, (2.57) becomes

$$\begin{aligned} \mathbf{f}_{j\mathbf{k}\to j'\mathbf{k}'}(\mathbf{q}) &= \frac{1}{V}\int \mathrm{d}^3x \left[\sum_{\mathbf{G}'} u^*_{j'}(\mathbf{k}'+\mathbf{G}')e^{-i(\mathbf{k}'+\mathbf{G}')\cdot\mathbf{x}}\right] e^{i\mathbf{q}\cdot\mathbf{x}}\frac{i\nabla_{\mathbf{x}}}{m_e} \left[\sum_{\mathbf{G}} u_j(\mathbf{k}+\mathbf{G})e^{i(\mathbf{k}+\mathbf{G})\cdot\mathbf{x}}\right] \\ &= -\frac{(2\pi)^3}{m_e V}\sum_{\mathbf{G}\mathbf{G}'} u^*_{j'}(\mathbf{k}'+\mathbf{G}')u_j(\mathbf{k}+\mathbf{G})\delta^{(3)}(\mathbf{k}-\mathbf{k}'+\mathbf{G}-\mathbf{G}'+\mathbf{q})(\mathbf{k}+\mathbf{G}), \end{aligned} \tag{2.60}$$

where we let the gradient act on the rightmost exponential in the second step.

To extract a Dirac delta from (2.58), we need to extract it from each of its three terms. To begin with, we have

$$\begin{aligned} |f_{j\mathbf{k}\to j'\mathbf{k}'}|^2 &= f_{j\mathbf{k}\to j'\mathbf{k}'}f^*_{j\mathbf{k}\to j'\mathbf{k}'} \\ &= \frac{(2\pi)^3}{V}\sum_{\mathbf{F}\mathbf{F}'} u^*_{j'}(\mathbf{k}'+\mathbf{F}')u_j(\mathbf{k}+\mathbf{F})\delta^{(3)}(\mathbf{k}-\mathbf{k}'+\mathbf{F}-\mathbf{F}'+\mathbf{q}) \\ &\times \frac{(2\pi)^3}{V}\left[\sum_{\mathbf{G}\mathbf{G}'} u^*_{j'}(\mathbf{k}'+\mathbf{G}')u_j(\mathbf{k}+\mathbf{G})\delta^{(3)}(\mathbf{k}-\mathbf{k}'+\mathbf{G}-\mathbf{G}'+\mathbf{q})\right]^*. \end{aligned} \tag{2.61}$$

Since we are summing over an infinite number of reciprocal lattice vectors $\mathbf{G}'$ and $\mathbf{F}'$, we can shift them so that we sum over $\Delta\mathbf{G} \equiv \mathbf{G}'-\mathbf{G}$ and $\Delta\mathbf{F} \equiv \mathbf{F}'-\mathbf{F}$ instead. Then

$$\begin{aligned} |f_{j\mathbf{k}\to j'\mathbf{k}'}|^2 &= f_{j\mathbf{k}\to j'\mathbf{k}'}f^*_{j\mathbf{k}\to j'\mathbf{k}'} \\ &= \frac{(2\pi)^6}{V^2}\sum_{\mathbf{F}\Delta\mathbf{F}} u^*_{j'}(\mathbf{k}'+\mathbf{F}+\Delta\mathbf{F})u_j(\mathbf{k}+\mathbf{F})\delta^{(3)}(\mathbf{k}-\mathbf{k}'+\mathbf{q}-\Delta\mathbf{F}) \\ &\times \sum_{\mathbf{G}\Delta\mathbf{G}} \left[u^*_{j'}(\mathbf{k}'+\mathbf{G}+\Delta\mathbf{G})u_j(\mathbf{k}+\mathbf{G})\right]^*\delta^{(3)}(\mathbf{k}-\mathbf{k}'+\mathbf{q}-\Delta\mathbf{G}) \\ &= \frac{(2\pi)^6}{V^2}\sum_{\Delta\mathbf{F}\Delta\mathbf{G}} \delta^{(3)}(\mathbf{k}-\mathbf{k}'+\mathbf{q}-\Delta\mathbf{F})\delta^{(3)}(\mathbf{k}-\mathbf{k}'+\mathbf{q}-\Delta\mathbf{G}) \\ &\times \sum_{\mathbf{F}} u^*_{j'}(\mathbf{k}'+\mathbf{F}+\Delta\mathbf{F})u_j(\mathbf{k}+\mathbf{F})\left[\sum_{\mathbf{G}} u^*_{j'}(\mathbf{k}'+\mathbf{G}+\Delta\mathbf{G})u_j(\mathbf{k}+\mathbf{G})\right]^*. \end{aligned} \tag{2.62}$$

We see from the two Dirac deltas that we only get non-zero terms when $\Delta\mathbf{F} = \Delta\mathbf{G}$, and by resolving the delta containing $\Delta\mathbf{F}$ we get a factor of $V/(2\pi)^3$. Furthermore, by defining [8]

$$f'_{j\mathbf{k}\to j'\mathbf{k}'} \equiv \sum_{\mathbf{G}} u^*_{j'}(\mathbf{k}' + \mathbf{G} + \Delta\mathbf{G}) u_j(\mathbf{k} + \mathbf{G}) \tag{2.63}$$

we see that

$$\begin{aligned} |f_{j\mathbf{k}\to j'\mathbf{k}'}|^2 &= \frac{(2\pi)^3}{V} \sum_{\Delta\mathbf{G}} \delta^{(3)}(\mathbf{k} - \mathbf{k}' + \mathbf{q} - \Delta\mathbf{G}) f'_{j\mathbf{k}\to j'\mathbf{k}'} (f'_{j\mathbf{k}\to j'\mathbf{k}'})^* \\ &= \frac{(2\pi)^3}{V} \sum_{\Delta\mathbf{G}} \delta^{(3)}(\mathbf{k} - \mathbf{k}' + \mathbf{q} - \Delta\mathbf{G}) |f'_{j\mathbf{k}\to j'\mathbf{k}'}|^2. \end{aligned} \tag{2.64}$$

Regarding the second term in (2.58), we observe in a similar way that

$$\begin{aligned} f_{j\mathbf{k}\to j'\mathbf{k}'} \mathbf{f}_{j\mathbf{k}\to j'\mathbf{k}'} &= -\frac{(2\pi)^6}{m_e V} \sum_{\mathbf{F}\mathbf{F}'} u^*_{j'}(\mathbf{k}' + \mathbf{F}') u_j(\mathbf{k} + \mathbf{F}) \delta^{(3)}(\mathbf{k} - \mathbf{k}' + \mathbf{F} - \mathbf{F}' + \mathbf{q}) \\ &\times \left[\sum_{\mathbf{G}\mathbf{G}'} u^*_{j'}(\mathbf{k}' + \mathbf{G}') u_j(\mathbf{k} + \mathbf{G}) \delta^{(3)}(\mathbf{k} - \mathbf{k}' + \mathbf{G} - \mathbf{G}' + \mathbf{q})(\mathbf{k} + \mathbf{G}) \right]^* \\ &= -\frac{(2\pi)^3}{m_e V} \sum_{\Delta\mathbf{G}} \delta^{(3)}(\mathbf{k} - \mathbf{k}' + \mathbf{q} - \Delta\mathbf{G}) \sum_{\mathbf{F}} u^*_{j'}(\mathbf{k}' + \mathbf{F} + \Delta\mathbf{G}) u_j(\mathbf{k} + \mathbf{F}) \\ &\times \left[\sum_{\mathbf{G}} u^*_{j'}(\mathbf{k}' + \mathbf{G} + \Delta\mathbf{G}) u_j(\mathbf{k} + \mathbf{G})(\mathbf{k} + \mathbf{G}) \right]^* \\ &= \frac{(2\pi)^3}{V} \sum_{\Delta\mathbf{G}} \delta^{(3)}(\mathbf{k} - \mathbf{k}' + \mathbf{q} - \Delta\mathbf{G}) f_{j\mathbf{k}\to j'\mathbf{k}'} (\mathbf{f}'_{j\mathbf{k}\to j'\mathbf{k}'})^*, \end{aligned} \tag{2.65}$$

where

$$\mathbf{f}'_{j\mathbf{k}\to j'\mathbf{k}'} \equiv -\frac{1}{m_e} \sum_{\mathbf{G}} u^*_{j'}(\mathbf{k}' + \mathbf{G} + \Delta\mathbf{G}) u_j(\mathbf{k} + \mathbf{G})(\mathbf{k} + \mathbf{G}). \tag{2.66}$$

Lastly, we rewrite the last term in (2.58). Using the same procedure of simplifying the sums, we obtain

$$\begin{aligned} |(\nabla_{\boldsymbol{\ell}} \mathcal{M})_{\boldsymbol{\ell}=0} \cdot \mathbf{f}_{j\mathbf{k}\to j'\mathbf{k}'}|^2 &= \left[(\nabla_{\boldsymbol{\ell}} \mathcal{M})_{\boldsymbol{\ell}=0} \cdot \mathbf{f}_{j\mathbf{k}\to j'\mathbf{k}'} \right] \left[(\nabla_{\boldsymbol{\ell}} \mathcal{M})_{\boldsymbol{\ell}=0} \cdot \mathbf{f}_{j\mathbf{k}\to j'\mathbf{k}'} \right]^* \\ &= -\frac{(2\pi)^3}{m_e^2 V} \sum_{\Delta\mathbf{G}} \delta^{(3)}(\mathbf{k} - \mathbf{k}' + \mathbf{q} - \Delta\mathbf{G}) \\ &\times (\nabla_{\boldsymbol{\ell}} \mathcal{M})_{\boldsymbol{\ell}=0} \cdot \sum_{\mathbf{F}} u^*_{j'}(\mathbf{k}' + \mathbf{F} + \Delta\mathbf{G}) u_j(\mathbf{k} + \mathbf{F})(\mathbf{k} + \mathbf{F}) \\ &\times \left[(\nabla_{\boldsymbol{\ell}} \mathcal{M})_{\boldsymbol{\ell}=0} \cdot \sum_{\mathbf{G}} u^*_{j'}(\mathbf{k}' + \mathbf{G} + \Delta\mathbf{G}) u_j(\mathbf{k} + \mathbf{G})(\mathbf{k} + \mathbf{G}) \right]^* \\ &= \frac{(2\pi)^3}{V} \sum_{\Delta\mathbf{G}} \delta^{(3)}(\mathbf{k} - \mathbf{k}' + \mathbf{q} - \Delta\mathbf{G}) \left| (\nabla_{\boldsymbol{\ell}} \mathcal{M})_{\boldsymbol{\ell}=0} \cdot \mathbf{f}'_{j\mathbf{k}\to j'\mathbf{k}'} \right|^2. \end{aligned} \tag{2.67}$$

Finally, we can use Eqs. (2.64), (2.65) and (2.67) in (2.58) to extract a Dirac delta in the scattering amplitude,

$$\begin{aligned}\overline{|\mathcal{M}_{j\mathbf{k}\to j'\mathbf{k}'}|^2} &= \frac{(2\pi)^3}{V}\sum_{\Delta\mathbf{G}}\delta^{(3)}(\mathbf{k}-\mathbf{k}'+\mathbf{q}-\Delta\mathbf{G}) \\ &\times\left(\overline{|\mathcal{M}|^2}\left|f'_{j\mathbf{k}\to j'\mathbf{k}'}\right|^2 + 2m_e\overline{\mathrm{Re}\left[\mathcal{M}f'_{j\mathbf{k}\to j'\mathbf{k}'}(\nabla_{\boldsymbol{\ell}}\mathcal{M})^*_{\boldsymbol{\ell}=0}\cdot(\mathbf{f}'_{j\mathbf{k}\to j'\mathbf{k}'})^*\right]}\right. \\ &\left.+ m_e^2\overline{|(\nabla_{\boldsymbol{\ell}}\mathcal{M})^2_{\boldsymbol{\ell}=0}\cdot\mathbf{f}_{j\mathbf{k}\to j'\mathbf{k}'}|^2}\right) \\ &\equiv \frac{(2\pi)^3}{V}\sum_{\Delta\mathbf{G}}\delta^{(3)}(\mathbf{k}-\mathbf{k}'+\mathbf{q}-\Delta\mathbf{G})\overline{\left|\mathcal{M}'_{j\mathbf{k}\to j'\mathbf{k}'}\right|^2}.\end{aligned} \tag{2.68}$$

This is the last property needed before proceeding with computing the transition rate.

2.2.3 The total electronic transition rate in crystals

We are now equipped to derive the total crystal transition rate in terms of the crystal form of the scattering amplitude developed in the last section. Equation (2.50) together with (2.27) and (2.68) leads to the total transition rate

$$\begin{aligned}\mathscr{R}_{\mathrm{crystal}} &= 2\sum_{jj'}\int_{\mathrm{BZ}}\frac{V\mathrm{d}^3k}{(2\pi)^3}\int_{\mathrm{BZ}}\frac{V\mathrm{d}^3k'}{(2\pi)^3}2\pi\delta(E_f-E_i)\frac{n_\chi}{16m_\chi^2m_e^2} \\ &\times\int\frac{\mathrm{d}^3q}{(2\pi)^3}\int d^3v f_\chi(v)\frac{(2\pi)^3}{V}\sum_{\Delta\mathbf{G}}\delta^{(3)}(\mathbf{k}-\mathbf{k}'+\mathbf{q}-\Delta\mathbf{G})\overline{\left|\mathcal{M}'_{j\mathbf{k}\to j'\mathbf{k}'}\right|^2}.\end{aligned} \tag{2.69}$$

To begin with, the energy Dirac delta can be simplified. In the non-relativistic limit, the eigenenergies of the initial and final states are given by the total energy of the system. This corresponds to the rest masses, the kinetic energy for the DM particle and the energy of the electron. With initial and final electron energy $E_{j\mathbf{k}}$ and $E_{j'\mathbf{k}'}$, we get

$$\begin{aligned}E_i &= m_\chi + m_e + \frac{\mathbf{p}^2}{2m_\chi} + E_{j\mathbf{k}} = m_\chi + m_e + \frac{m_\chi v^2}{2} + E_{j\mathbf{k}}, \\ E_f &= m_\chi + m_e + \frac{\mathbf{p}'^2}{2m_\chi} + E_{j'\mathbf{k}'} = m_\chi + m_e + \frac{|m_\chi\mathbf{v}-\mathbf{q}|^2}{2m_\chi} + E_{j'\mathbf{k}'},\end{aligned} \tag{2.70}$$

where $v = |\mathbf{v}|$ is the initial speed of the DM particle. We can also introduce a relative angle θ_{qv} between $\mathbf{v}$ and the transferred momentum $\mathbf{q}$ such that

$$\begin{aligned}E_f - E_i &= \Delta E_{j\mathbf{k}\to j'\mathbf{k}'} + \frac{|m_\chi\mathbf{v}-\mathbf{q}|^2}{2m_\chi} - \frac{m_\chi v^2}{2} \\ &= \Delta E_{j\mathbf{k}\to j'\mathbf{k}'} + \frac{q^2}{2m_\chi} - \mathbf{v}\cdot\mathbf{q} \\ &= \Delta E_{j\mathbf{k}\to j'\mathbf{k}'} + \frac{q^2}{2m_\chi} - qv\cos\theta_{qv},\end{aligned} \tag{2.71}$$

where $\Delta E_{j\mathbf{k}\to j'\mathbf{k}'} \equiv E_{j'\mathbf{k}'} - E_{j\mathbf{k}}$ and $q = |\mathbf{q}|$. Now, by reorganising (2.69) and inserting (2.71), we find

$$\begin{aligned}\mathscr{R}_{\text{crystal}} = \frac{\pi V n_\chi}{4m_\chi^2 m_e^2} \int \mathrm{d}^3q \int_{\text{BZ}} \frac{\mathrm{d}^3k}{(2\pi)^3} \int_{\text{BZ}} \frac{\mathrm{d}^3k'}{(2\pi)^3} \sum_{jj'\Delta\mathbf{G}} \delta^{(3)}\left(\mathbf{k}-\mathbf{k}'+\mathbf{q}-\Delta\mathbf{G}\right) \\ \times \int d^3v f_\chi(\mathbf{v}) \delta\left(\Delta E_{j\mathbf{k}\to j'\mathbf{k}'} + \frac{q^2}{2m_\chi} - qv\cos\theta_{qv}\right) \overline{\left|\mathcal{M}'_{j\mathbf{k}\to j'\mathbf{k}'}\right|^2}.\end{aligned} \tag{2.72}$$

If we define [8]

$$\xi = \frac{q}{2m_\chi v} + \frac{\Delta E_{j\mathbf{k}\to j'\mathbf{k}'}}{qv}, \tag{2.73}$$

we can also write

$$\delta\left(\Delta E_{j\mathbf{k}\to j'\mathbf{k}'} + \frac{q^2}{2m_\chi} - qv\cos\theta_{qv}\right) = \delta(qv(\cos\theta-\xi)) = \frac{1}{qv}\delta(\cos\theta-\xi). \tag{2.74}$$

For the scattering process to occur we must have $v \neq 0$, and from the leftmost energy Dirac delta in (2.74) we see that $q = 0$ would imply that there is no electronic transition. Thus, it is safe to divide by qv.

We continue by evaluating the velocity integral in (2.72) [8]. With polar coordinates,

$$\int \mathrm{d}^3v = \int \mathrm{d}v v^2 \int_0^{2\pi} \mathrm{d}\phi \int_{-1}^{1} \mathrm{d}\cos\theta, \tag{2.75}$$

the velocity integral can be written as

$$\Gamma_{j\mathbf{k}\to j'\mathbf{k}'} \equiv \int \mathrm{d}v v \int_0^{2\pi} \mathrm{d}\phi \int_{-1}^{1} \mathrm{d}\cos\theta f_\chi(\mathbf{v}) \frac{1}{q} \delta(\cos\theta-\xi) \overline{\left|\mathcal{M}'_{j\mathbf{k}\to j'\mathbf{k}'}\right|^2}. \tag{2.76}$$

In general, the velocity distribution $f_\chi(\mathbf{v})$ is not spherically symmetric, but for simplicity, we now approximate $f_\chi(\mathbf{v}) = f_\chi(v)$ [12, App. A.2]. Then it is possible to use the Dirac delta to evaluate the integral over θ such that

$$\Gamma_{j\mathbf{k}\to j'\mathbf{k}'} \approx \int_{v\geq v_{\text{min}}} \mathrm{d}v v^2 \frac{f_\chi(v)}{v} \int_0^{2\pi} \mathrm{d}\phi \overline{\left|\mathcal{M}'_{j\mathbf{k}\to j'\mathbf{k}'}\right|^2}_{\cos\theta=\xi}. \tag{2.77}$$

As a consequence, the lower limit of the velocity integral was moved up from 0 to $v_{\text{min}} \equiv \xi v$, since $v \geq v_{\text{min}}$ ensures that $\cos\theta = \xi = v_{\text{min}}/v \leq 1$. As in Reference [8], we define the squared transition amplitude averaged over the azimuthal angle,

$$\overline{\overline{\left|\mathcal{M}'_{j\mathbf{k}\to j'\mathbf{k}'}\right|^2}}_{\cos\theta=\xi} \equiv \frac{1}{2\pi} \int_0^{2\pi} \mathrm{d}\phi \overline{\left|\mathcal{M}'_{j\mathbf{k}\to j'\mathbf{k}'}\right|^2}_{\cos\theta=\xi}, \tag{2.78}$$

so that

$$\Gamma_{j\mathbf{k}\to j'\mathbf{k}'} \approx 2\pi \int_{v\geq v_{\text{min}}} \mathrm{d}v v^2 \frac{f_\chi(v)}{v} \overline{\overline{\left|\mathcal{M}'_{j\mathbf{k}\to j'\mathbf{k}'}\right|^2}}_{\cos\theta=\xi}. \tag{2.79}$$

Moreover, we introduce the operator $\hat{\eta}$ which acts as

$$\hat{\eta}\left[Cg(\mathbf{v})\right] = C \int_{|\mathbf{v}|\geq v_{\text{min}}} \mathrm{d}^3v \frac{f_\chi(\mathbf{v})}{v} g(\mathbf{v}) \tag{2.80}$$

on a constant C and an arbitrary function $g(\mathbf{v})$ of the incoming DM-velocity [8]. If $g(\mathbf{v})$ and $f_\chi(\mathbf{v})$ were isotropic, then spherical coordinates lead to

$$\hat{\eta}\left[Cg(\mathbf{v})\right] = 4\pi C \int_{v\geq v_{\min}} \mathrm{d}v v^2 \frac{f_\chi(v)}{v} g(v) \tag{2.81}$$

and we can write

$$\Gamma_{j\mathbf{k}\to j'\mathbf{k}'} \approx \frac{1}{2}\hat{\eta}\left[\overline{\overline{\left|\mathcal{M}'_{j\mathbf{k}\to j'\mathbf{k}'}\right|^2}}_{\cos\theta=\xi}\right] \tag{2.82}$$

Inserting this approximation of the velocity integral in (2.72), we find that

$$\begin{aligned}\mathscr{R}_{\text{crystal}} = \frac{\pi V n_\chi}{8 m_\chi^2 m_e^2} \int_{\text{BZ}} \frac{\mathrm{d}^3 k}{(2\pi)^3} \int_{\text{BZ}} \frac{\mathrm{d}^3 k'}{(2\pi)^3} \sum_{jj'\Delta\mathbf{G}} \\ \times \int \mathrm{d}^3 q \frac{1}{q} \delta^{(3)}\left(\mathbf{k}-\mathbf{k}'+\mathbf{q}-\Delta\mathbf{G}\right) \hat{\eta}\left[\overline{\overline{\left|\mathcal{M}'_{j\mathbf{k}\to j'\mathbf{k}'}\right|^2}}_{\cos\theta=\xi}\right].\end{aligned} \tag{2.83}$$

The last step is to involve the effective theory description of the scattering amplitude.

2.2.4 The dark matter and crystal response functions

Equipped with the expression for the electronic transition rate in crystal detectors in (2.83), we are finally able to connect the transition rate to the effective theory presented in Section 2.1.4. Inspired by (2.83), let us define the l-th crystal response functions [8]

$$\begin{aligned}W_l(\mathbf{q},\Delta E) = (4\pi)^2 V_{\text{cell}} \Delta E \sum_{jj'\Delta\mathbf{G}} \int_{\text{BZ}} \frac{\mathrm{d}^3 k}{(2\pi)^3} \int_{\text{BZ}} \frac{\mathrm{d}^3 k'}{(2\pi)^3} B_l \\ \times \delta^{(3)}(\mathbf{k}-\mathbf{k}'+\mathbf{q}-\Delta\mathbf{G}) \delta(\Delta E_{j\mathbf{k}\to j'\mathbf{k}'} + \Delta E),\end{aligned} \tag{2.84}$$

where

$$\Delta E \equiv \frac{q^2}{2m_\chi} - qv\cos\theta_{qv}. \tag{2.85}$$

and B_l are terms containing $f'_{j\mathbf{k}\to j'\mathbf{k}'}$ and $\mathbf{f}'_{j\mathbf{k}\to j'\mathbf{k}'}$; see [8, (38 − 42)]. The factor of ΔE outside the integrals makes W_l dimensionless.

Because of the structure of the crystal response functions, it is apparent that they can replace parts of (2.83). In fact, it can be shown that the transition rate can be written as a sum of products between the crystal response functions and the DM response functions, R_l, such that [8]

$$\mathscr{R}_{\text{crystal}} = \frac{n_\chi N_{\text{cell}}}{128\pi m_\chi^2 m_e^2} \int \mathrm{d}(\ln \Delta E) \int \mathrm{d}^3 q \frac{1}{q} \hat{\eta}(q,\Delta E) \sum_{l=1}^{r} \mathrm{Re}\left[R_l^*(q,v) W_l(\mathbf{q},\Delta E)\right] \tag{2.86}$$

The (dimensionless) DM response functions are given in Reference [8, App. C], and are functions of the EFT coupling strengths c_i introduced in (2.45). This thesis concerns only silicon and germanium target materials, for which the relevant number of response functions is $r = 5$ [8]. It is out of the scope of this thesis to discuss the

crystal and DM response functions in detail. Instead, it is emphasised that the transition rate can be written in terms of the response functions in this way and that this is how the transition rate depends on the EFT.

With the transition rate expressed in this way, we can continue by evaluating the integral over transferred momenta, $\mathbf{q}$. By using spherical coordinates for $\mathbf{q}$ as in (2.75), we find

$$\begin{aligned}\mathscr{R}_{\text{crystal}} = \frac{n_\chi N_{\text{cell}}}{128\pi m_\chi^2 m_e^2} \int \mathrm{d}(\ln \Delta E) \\ \times \int \mathrm{d}q q \hat{\eta}(q, \Delta E) \sum_{l=1}^{r} \mathrm{Re}\left[R_l^*(q, v)\overline{W}_l(\mathbf{q}, \Delta E)\right],\end{aligned} \tag{2.87}$$

where

$$\begin{aligned}\overline{W}_l(\mathbf{q}, \Delta E) &\equiv \int \mathrm{d}\Omega W_l(\mathbf{q}, \Delta E) \\ &= \int \mathrm{d}\Omega \int \mathrm{d}q' W_l(\mathbf{q}', \Delta E)\delta(q' - q) \\ &= \frac{1}{q^2} \int \mathrm{d}^3 q' W_l(\mathbf{q}', \Delta E)\delta(|\mathbf{q}'| - q).\end{aligned} \tag{2.88}$$

We introduced a new integral over $|\mathbf{q}'|$ to be able to simplify the three-dimensional Dirac delta in (2.84), since

$$\int \mathrm{d}^3 q' \delta^{(3)}(\mathbf{k} - \mathbf{k}' + \mathbf{q}' - \Delta\mathbf{G})\delta(|\mathbf{q}'| - q) = \delta(|\mathbf{k} - \mathbf{k}' - \Delta\mathbf{G}| - q). \tag{2.89}$$

This leads to

$$\begin{aligned}\overline{W}_l(\mathbf{q}, \Delta E) = \frac{(4\pi)^2 V_{\text{cell}} \Delta E}{q^2} \sum_{jj'\Delta\mathbf{G}} \int_{\text{BZ}} \frac{\mathrm{d}^3 k}{(2\pi)^3} \int_{\text{BZ}} \frac{\mathrm{d}^3 k'}{(2\pi)^3} B_l \\ \times \delta(|\mathbf{k} - \mathbf{k}' - \Delta\mathbf{G}| - q)\delta(\Delta E_{j\mathbf{k}\to j'\mathbf{k}'} + \Delta E).\end{aligned} \tag{2.90}$$

Equation (2.87) is the final expression for the crystal responses that is used in the numerical implementations.

2.2.5 Electron-hole pair formalism

The final transition rate is given by (2.87). The aim of this project is to develop a GPU program and to model this transition rate by implementing a neural network, and in these ways accelerate the computations of the transition rate. The quantity that will be studied is the transition rate as a function of the number of electron-hole pairs. When an electron transitions from the valence band to the conduction band, an electron-hole pair appears. Consequently, the number of electron-hole pairs, Q, depends on the available energy, ΔE, in the scattering process [8],

$$Q = 1 + \lfloor (\Delta E - E_{\text{gap}})/\varepsilon \rfloor \tag{2.91}$$

as well as the tabulated values of the valence-conduction band gap, E_{gap}, and the required energy to create an electron-hole pair for the specific target material, ε. In our case, the target material is either silicon or germanium.

Since the number of electron-hole pairs is a floor function of the deposited energy, there is a continuum of energies that corresponds to the same number of electron-hole pairs. Thus, the transition rate as a (discrete) function of the number of electron-hole pairs corresponds to integrating over the range of ΔE that leads to the given Q. This is the quantity that will result from the neural network.

3
Parallelising the computations

The aim of this thesis is to accelerate the numerical computations of the electronic transition rate that was derived in Chapter 2. In this thesis, it is done both with parallelisation and with ANNs. This chapter describes the general idea for computing the transition rates, as well as the procedure for the speed improvements developed in this thesis. The full repository for the implementations presented in this thesis is published on Github, at HannaOlvhammar/Accelerating_DM_computations. The Python program from Section 3.1 is in the directory named "original", the unparallelised C++ program from Section 3.2 is in the "cpp_cpu" directory, and the GPU program from Section 3.3.2 is in the "cuda" directory.

3.1 The data generation program

To compute the electronic transition rates numerically, (2.87) is used. The integrals in (2.87) are computed as Riemann sums, which, in a computer program, translate to for-loops that sum over the integral variables. The first program that was created for computing the electronic transition rates was written in Python[1] and is referred to as the Python program in this thesis. While both the programming language and general structure of this program are experimented with in this thesis, the idea for computing the transition rate remains. Algorithm 1 shows the pseudo-code for the computations common to all of the programs studied in this thesis and is based on (2.87) and associated sections.

The crystal response functions, $\overline{W}_l$, the integrated range of the transferred momentum, q, and the range of transition energies, ΔE, were all generated in separate files which have remained unaltered in this thesis. The outputs of those files are read and used in the computations of the integrals included in the transition rate, $\mathscr{R}_{\text{crystal}}$.

The electronic transition rates are stored in a matrix, $\mathscr{R}_{\text{crystal}}$, with n_D rows and n_Q columns and all elements initialised to zero. The outermost for-loop in Algorithm 1 loops over all n_D input data samples, such that for each set of input parameters one row in the transition rate matrix is computed. When the for-loop over the transition energies, ΔE, is performed, the number of possible electron-hole pairs, Q, is computed. Based on the result, the corresponding column in the transition rate matrix is computed. Inside all for-loops and if-statements, the integrand in (2.87) is computed and added to the corresponding element in the matrix $\mathscr{R}_{\text{crystal}}$. This

[1]The program was written by my co-supervisor Einar Urdshals.

Algorithm 1 The pseudo-code for generating transition rates given a set of inputs

1: **for** $i_D < n_D$ **do** ▷ For each data sample
2: **for** all ΔE in pre-generated range **do** ▷ $\int \mathrm{d}(\Delta E)$
3: $Q \leftarrow 1 + \lfloor (\Delta E - E_{\mathrm{gap}})/\varepsilon \rfloor$ ▷ (2.91)
4: **if** $Q < n_Q$ **then** ▷ Possible transitions for given Q
5: **for** all q in pre-generated range **do** ▷ $\int q$
6: $v_{\mathrm{min}} \leftarrow \Delta E/q + q/2m_\chi$ ▷ $v_{\mathrm{min}} = \xi v$, see (2.73)
7: **if** $v_{\mathrm{min}} < v_{\mathrm{esc}} + v_\oplus$ **then**
8: Call functions for computing $R_l(q, v)$ and $\hat{\eta}(q, \Delta E)$
9: $x \leftarrow q\hat{\eta}(q, \Delta E) \sum_{l=1}^{r} \mathrm{Re}\left[R_l^*(q, v)\overline{W}_l(\mathbf{q}, \Delta E)\right]$ ▷ (2.87)
10: $\mathscr{R}_{\mathrm{crystal}}[i_D, Q] \leftarrow \mathscr{R}_{\mathrm{crystal}}[i_D, Q] + x \cdot \mathrm{constants}$
11: **end if**
12: **end for**
13: **end if**
14: **end for**
15: **end for**

way, each iteration of the for-loops adds a term to the sum for the specific matrix element, performing the Riemann sums that replace the integrals in (2.87).

The input parameters in this thesis are either the DM mass, m_χ, which enters in the constants on line 12 of Algorithm 1, or the EFT coupling strengths which enter in the DM response functions, R_l.

3.2 From Python to C++

The primary aim of this thesis was to accelerate numerical computations for DM research using ANNs. To develop the ANN, large amounts of training data in different formats needed to be generated. To speed up the development process of the ANN, the Python program for generating data could be optimised.

It became apparent that the Python program did not utilise any tools that were specific to Python or were more convenient and readable in Python compared to C++. Python libraries such as Scikitlearn and SciPy can be very valuable in scientific programming, but the only significant library used in the Python program was NumPy. NumPy is a numerical library for Python that is written in optimised C code, and all NumPy functions used in the Python program are accessible as C functions [19]. Therefore, the Python program could easily be rewritten in C++.

There are several reasons that C++ is faster than Python in general. One is that Python is an interpreted language while C++ is a compiled language. This means that each time a Python program runs, the code must be parsed, interpreted and executed [20]. In contrast, a C++ program needs to be compiled once to produce an executable file. The executable file is then written in machine language that very efficiently runs on the computer it was compiled on, as many times as needed. Another reason is that Python types are stored as objects, which can make e.g. a large array of floats consume a lot more memory, and thus time, than a C++ array [21]. Using the NumPy library can prevent this sort of overhead, however.

One way to speed up the Python program is to use just-in-time compilation, which is provided by e.g. the library Numba. Numba provides a decorator for Python functions such that the first time a function with a just-in-time decorator is called, Numba compiles the function and translates it into machine code [22]. The machine code is then executed every time the function is called in the program. Numba was used for all functions in the original Python program with the purpose of speeding up the computations without changing the overall structure of the program.

It would seem that a Python program using Numba would be as fast as a C++ program, but this is not always the case. Numba can prove to be faster than naively implemented C++ code, which in our case could mean a line-by-line translation of the Python code into C++. However, optimised C++ code with advanced compiler options for optimisation can be faster than Python code with Numba functions [23]. This motivated an attempt to write the Python program in C. While C could be used instead of C++ in this thesis, some functions such as probability distributions and array handling are more convenient to use in C++.

In the C++ program, the overall structure for computing the electronic transition rates in Algorithm 1 remained. However, the code was optimised in several ways. Some examples include decreasing the number of function calls, removing repeated computations and decreasing the number of involved files. These improvements could also have been implemented in the Python program. What could not be done as easily in Python was to optimise memory usage, which in C++ consisted of carefully allocating and freeing memory for the necessary arrays. Furthermore, the O3 optimisation compiler flag was used in the C++ program, providing further automated optimisation [24].

The most important improvement in the structure of the program was to vectorise computations. Vectorising computations can be described as the difference between a for-loop that calls a function in each iteration and a single call to a function that contains the for-loop. The latter is a vectorised function, which not only provides a speed gain by decreasing the number of function calls; the compiler can optimise the execution of the function so that each iteration is run in parallel on the central processing unit (CPU) [25].

Writing the Python program in C++ made the generation of the electronic transition rates about twice as fast; see Section 5.3. Having already vectorised computations in C++, the possibility for further seed-ups was recognised in the form of GPU computations.

3.3 The GPU program

There are ways to improve the performance of the data generation programs running on the CPU, such as optimising the program and writing it in C++, as was discussed in Section 3.2. In this section, the method of using GPUs to accelerate computations is presented. First, the principles of parallel computing are reviewed, and then the GPU implementation for accelerating the computation of transition rates is presented.

3.3.1 Parallelisation principles

Both the CPU and GPU are used to perform computations for different purposes. The CPU is specialised in executing fast sequential instructions from e.g. the operating system or the calculator; the CPU has a high clock speed. The GPU is mainly used for parallel instructions, such as rendering thousands of polygons simultaneously in a computer game. In Section 3.1, the Python program runs on the CPU. This means that each iteration of the for-loop is fast, but that the instructions are executed sequentially. If the iterations were instead executed in parallel, the for-loop could be performed faster. Naturally, this requires that the iterations are independent.

The execution of an instruction is performed on a so-called thread. In a non-parallel program, all computations are executed on the same thread. In parallel computations, several threads are used to execute the instructions in parallel. It is possible to parallelise computations on the CPU, but even though the CPU has access to tens of threads, the GPU has access to thousands of threads [26]. This means that the GPU has a much higher capability of speeding up large for-loops with parallelisation. On the other hand, it has a lower clock speed than the CPU.

A program can be parallelised by importing tools from libraries such as OpenCL or CUDA [27, 26]. In this thesis, CUDA is used. Computations can be performed on the GPU by communicating with the GPU from the CPU. In CUDA, this is done through the following procedure:

1. Allocate arrays on the CPU (e.g. with `malloc()`).
2. Initialise the CPU arrays to their desired values.
3. Allocate arrays on the GPU (with `cudaMalloc()`).
4. Copy the contents of the CPU arrays to the GPU arrays (with `cudaMemcpy()`).
5. Call the GPU function and execute instructions in parallel on the GPU.
6. Copy the contents of the GPU arrays to the CPU arrays (with `cudaMemcpy()`).
7. Use the CPU arrays to e.g. save data in a separate file.

Allocating memory and copying data is computationally expensive, and provides a time overhead for the program. Consequently, the GPU program is not faster than the CPU program until a large enough amount of GPU threads are utilised.

There are some requirements for running computations on the GPU. First, to be able to run the relevant functions on the GPU they need to be either possible to call from the CPU, or callable only from the GPU. Such functions require the `__global__` or `__device__` declaration specifier, respectively [26]. Second, all of the GPU functions need to be gathered in a separate CUDA file marked with the `.cu` extension. Third, all of the relevant files need to be compiled with the CUDA compiler, `nvcc` [26]. Lastly, the code needs to be restructured both on the CPU-end to be able to call the GPU functions, and from the GPU-end to be able to assign for-loop iterations to different threads. The procedure for the CPU is mostly described with the memory transfer process above.

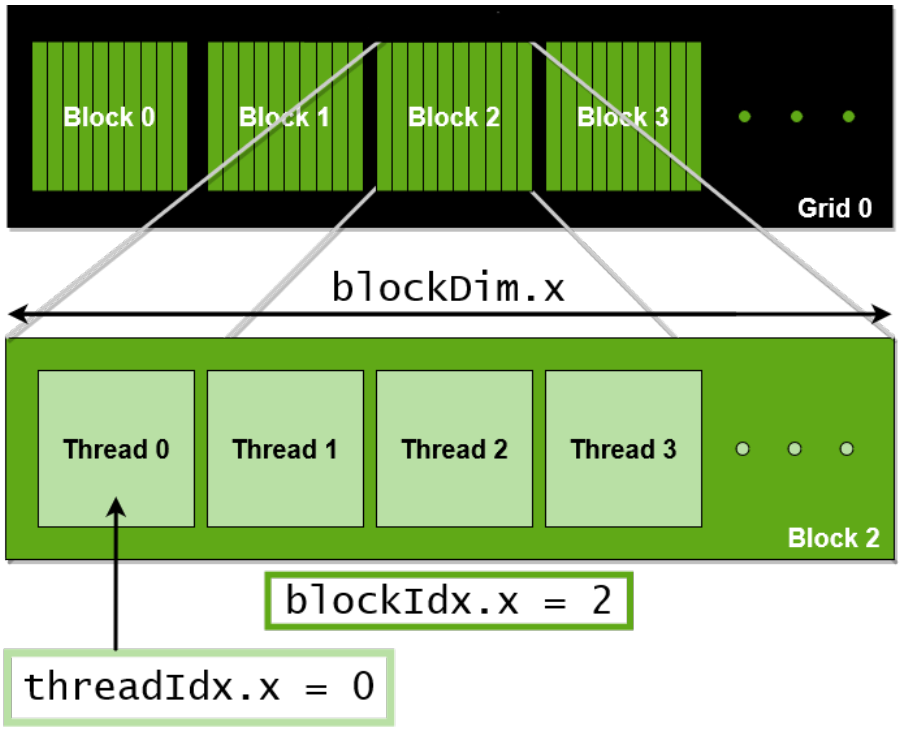


Figure 3.1: An illustration of the GPU software architecture [28].

For the GPU, the code needs to be executed simultaneously on different threads. The threads can be either one-, two- or three-dimensional and are organised in thread blocks, which in turn can have up to three dimensions and are organised in so-called grids. The grids can also be up to three-dimensional. Figure 3.1 illustrates the structure of the threads in one dimension. When using CUDA for GPU parallelisation, variables such as the thread index, block index and block size (also called block dimension) for each of the three dimensions are available to the thread that is executing the instruction. These variables can then be used in the code to assign different iterations of a for-loop to different threads. A practical example of how this type of code can look is given in Section 3.3.2, in the context of computing the electronic transition rates.

3.3.2 The electronic transition rates with GPUs

When accelerating the computation of electronic transition rates that was presented in Section 3.1, only the for-loop over data points was altered in Algorithm 1. This loop usually requires more iterations than the loops over transition energy and momentum and was thus deemed best suited for parallelisation as a first step.

Algorithm 2 contains the GPU equivalent of the data sample for-loop in Algorithm 1 for one-dimensional threads. In line 1 of Algorithm 2, the value of the iteration variable i_D is determined by which thread is executing the lines. Based on the block index, the size of the block and the index of the thread in that block, the iteration variable for the loop is determined, similar to how a matrix element is accessed in a row-major array. Compare to the layout in Figure 3.1. Line 2 in Algorithm 2 ensures that the iteration variable is smaller than n_D, to handle cases

Algorithm 2 The GPU equivalent of line 1 in Algorithm 1

1: $i_D \leftarrow$ blockIdx.x $\cdot$ blockDim.x + threadIdx.x ▷ Uses GPU variables
2: **if** $i_D < n_D$ **then** ▷ Replaces the for-loop
3: ...
4: **end if**

where there are more available threads than iterations.

The speed-up given by using parallelisation on GPUs is presented in Chapter 5. However, since the concept of multidimensional threads and blocks has been presented in this chapter and Algorithm 1 contains three for-loops, it is clear that the program can be further accelerated by utilising more dimensions of the GPU software. This type of layout was explored in this thesis but because of time constraints, it is instead suggested as an extension of this program.

The GPU program was first developed to accelerate the generation of data used for training the ANN presented in Chapter 4. However, the GPU program as a standalone program has proven to be a good alternative method for accelerating the computation of the electronic transition rates, independent of using ANNs. The transition rates computed in the GPU program are equal to the ones computed by the Python program. Furthermore, the GPU program can compute transition rates corresponding to all 28 EFT coupling strengths without any increment in execution time; it has been written in a way that transfers all EFT coupling strengths from the CPU to the GPU regardless of whether most or none of them are initialised to zero. This will turn out to be an advantage compared to the ANN.

4

Developing the artificial neural network

The purpose of this thesis is to accelerate the computations of DM-induced electronic transition rates in crystal detectors. The underlying theory of this process was discussed in Chapter 2, and a numerical implementation based on GPUs was presented in Chapter 3. In this chapter, the method of training and improving an ANN is discussed.

This chapter starts with an introduction to the theory of ANNs, specifically with a review of the concepts and tools that were needed to build the ANN in this thesis. Next, the ANN developed for predicting the transition rates is discussed and motivated. Chapter 5 provides the results for the ANN model presented in this chapter, and further motivates some choices of the setup in the context of desirable results. The files used to train and analyse the ANN presented in this chapter can be found on Github, at HannaOlvhammar/Accelerating_DM_computations in the "ann" directory.

4.1 The theory of artificial neural networks

Before developing the ANN, it is important to understand the theory of neural networks. This section reviews the most important concepts needed for developing and improving an ANN for a regression problem, but are not specific to the problem of computing transition rates.

4.1.1 The neural network layout

Inspired by the neural networks in the human brain, ANNs consist of a number of connected neurons. In science, ANNs are used to create models. The idea is that the ANN can learn and improve itself when training on new information, analogously to how neurons can make new connections in the brain.

Let $s_i(t)$ denote the state of a neuron i at a time step t, such that it indicates the level of activity for the neuron [29]. This neuron can then be connected to other neurons in a neural network, where the strength of the connection from neuron i to neuron j is measured by the weight w_{ij}. The state of a neuron at time step $t+1$ is then updated based on its connection to other neurons via an activation function, g,

$$s_i(t+1) = g(b_i), \tag{4.1}$$

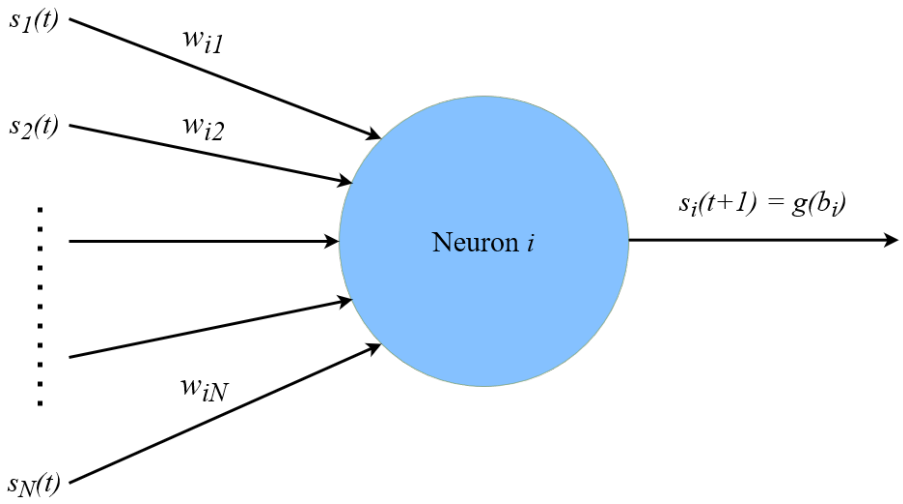


Figure 4.1: A schematic of a single neuron, indexed with i, in an ANN. It receives inputs from N other neurons at time step t, each with their assigned weights. The weights are then used to compute the output of neuron i, which becomes the input for other neurons in the network at time step $t + 1$.

where b is a local field defined as

$$b_i = \sum_{j=1}^{N} w_{ij} s_j(t) - \theta_i, \tag{4.2}$$

and $-\theta_i$ are biases. See Figure 4.1 [29].

The neurons in an ANN are placed in layers. In a feed-forward ANN, the states and weights of the neurons in the first layer of the ANN are the ones that determine the states of the neurons in the second layer via the activation function. The most simple form of an ANN consists of one input layer and one output layer. The number of neurons in the input layer is simply the number of input parameters for the model, and the number of neurons in the output layer is the number of outputs from the model. The neuron states $s_j(t)$ in the first layer will then correspond to the input values in the data.

A deep ANN has one or more layers of neurons between the input and output layers, so-called hidden layers. If there are two hidden layers, they can be thought of as modelling two functions $f^{(1)}$ and $f^{(2)}$ such that the ANN in total models a function $f = f^{(2)}(f^{(1)})$, see Figure 4.2 [30].

When an ANN is training, or learning, it repeatedly sends data through the neural network layers, assesses the performance of the model, and processes the data again. When all the data has been processed once by the neural network, one epoch has been performed.

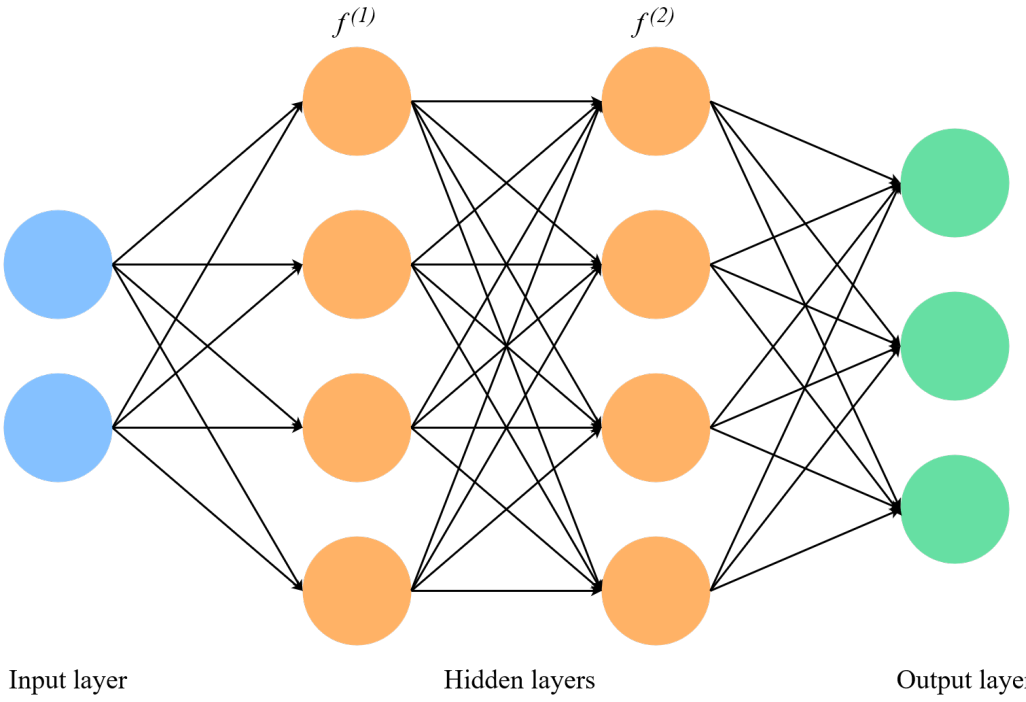


Figure 4.2: An example of the layers of an ANN. It has one input layer, two hidden layers and one output layer. The hidden layers model two functions $f^{(1)}$ and $f^{(2)}$ such that the total model of the ANN becomes $f = f^{(2)}(f^{(1)})$.

4.1.2 Training the neural network

To be able to train an ANN, one first needs to define the requirements for a good model. In regression problems, the most common way to quantify the accuracy of a model is with the mean squared error (MSE),

$$L_{\text{MSE}}^{(k)} = \frac{1}{n_D} \sum_{i=1}^{n_D} \left(y_i^{(k)} - O_i^{(k)} \right)^2, \tag{4.3}$$

where $O_i^{(k)}$ are the outputs predicted by the ANN after epoch k for n_D data samples. The $y_i^{(k)}$ are the data samples of the output parameter, commonly called labels; the input parameters are commonly called features. The loss function is computed after each epoch and is used to optimise the input weights for the following epoch. The loss function in (4.3) depends on the weights w_{ij} through the outputs $O_i^{(k)}$, and therefore the loss function is minimised by adjusting the weights accordingly [29].

The minimisation is usually performed with gradient descent methods such that new weights are updated by using the previous weight w_{ij} according to

$$w'_{ij} = w_{ij} + \delta w_{ij}, \tag{4.4}$$

where the descent step is given by

$$\delta w_{ij} = -\eta \frac{\mathrm{d}L_{\text{MSE}}}{\mathrm{d}w_{ij}} \tag{4.5}$$

and the learning rate η is set when initialising the ANN [29]. It is common to use stochastic gradient descent for the minimisation step, as is done in this project, to decrease the risk of getting stuck in local minima. In the case of stochastic gradient descent, the sum over samples in (4.3) is removed; instead, one sample is randomly selected for each of the weight updates. This also means that stochastic gradient descent is significantly faster. While the loss function does not necessarily decrease in a single iteration, the sample average of δw_{ij} is still negative [29].

4.1.3 Preprocessing the data

It is recommended to preprocess the input data and, if necessary, the output data for the ANN. Most importantly, shifting and scaling the input data to a zero mean and unit variance usually makes the ANN perform better [29]. While preprocessing can remove information from the data, shifting and scaling the data leads to better performance for the ANN in general. One reason for this is that large mean values and variances in the input data can lead to a steep gradient for the loss function. We also see from the local field in (4.2) that large input values can lead to large local fields, which in turn can lead to saturated neurons in the layer following the input layer.

Furthermore, some ANN implementations may assume that the data is distributed in a standard way. For example, L_1 and L_2 regularisation methods can assume that the labels are distributed with zero mean and unit variance [31].

4.1.4 Preventing overfitting

When an ANN leads to a model that performs well only on the data set that it has trained on, the ANN is said to have overfitted. This happens when the ANN gives too much importance to specific data samples and not the overall behaviour of the data set, and can be caused by e.g. an excessive amount of neurons in the ANN.

One way to prevent overfitting is to use cross-validation, where the data set is split into a training set and a validation set [29]. The ANN is only trained on the training set, while the validation set is used for checking the performance of the ANN on data it has not been trained on. In particular, the loss function (see Section 4.1.2) can be monitored for both the training set and the validation set during training. Overfitting then becomes apparent when the loss function starts increasing for the validation set while it is decreasing for the training set. Monitoring the loss function in this way provides a good basis for the appropriate number of epochs for the training of the ANN, since the training can be stopped before the validation loss starts increasing.

Another way to prevent overfitting is with regularisation. Regularisation schemes typically add terms to the loss function for the training set, which can lead to the training loss being larger than the validation loss. This is important to be aware of when comparing the training and validation loss. One example of a regularisation method is the use of drop-out, where a certain fraction of neuron weights in each hidden layer is set to zero, while the rest of the neurons are updated as usual. The neurons that are ignored vary in each step of the ANN. The reason for using drop-

out is that it simulates averaging the results of several different networks, which has been shown to prevent overfitting [29]. Other examples of regularisation methods are pruning, weight decay and batch normalisation, but they are not explored in this thesis [29].

4.2 Implementing the neural network

With the concepts presented in Section 4.1, we are equipped to design and implement an ANN. Since the purpose of the ANN is to speed up the computation of electronic transition rates, this section starts with connecting the transition rate formalism of Chapter 2 to the theory of ANNs discussed in Section 4.1. Then, the preprocessing of the data and the setup of the ANN developed in this thesis is presented and motivated. While this chapter connects more to the theoretical description of ANNs, Chapter 5 provides examples of desirable results for the ANN and further motivates some choices of the ANN design.

4.2.1 Electronic transition rates as a neural network problem

In this report, the DM-induced electronic transition rates in detector crystals are studied. For this purpose, the DM mass, m_χ, and the EFT coupling strengths c_7^s and c_7^l were chosen as the input parameters, while the output data is the transition rates for a given number of electron-hole pairs in the crystal. A neural network with only m_χ as the input parameter was also developed, with $c_7^s = 1$ and the rest of the EFT coupling strengths set to zero. Mainly the neural network with three input parameters is discussed in this chapter, but the results of both the one-input and the three-input ANNs are presented in Chapter 5.

In previous works, the $\mathcal{O}_1$ operator has been widely studied because of its simplicity. Here, the EFT is further investigated by studying the $\mathcal{O}_7$ coupling strengths, since it is clear from (2.47) that the $\mathcal{O}_7$ interactions have a relatively simple relationship between the spin and velocity properties of the scattering process. In the ideal case all 28 EFT coupling strengths from Section 2.1.4 would be included, but by limiting the number of input parameters the ANN is more easily analysed and improved as a first step. See Section 6.2 for further discussion. Since the transition rates for up to $n_Q = 10$ electron-hole pairs are studied, there are also n_Q output parameters. This constitutes a regression problem, where the ANN produces a model $f : \mathbb{R}^{n_p} \to \mathbb{R}^{n_Q}$ [30]. In this case, there are $n_p = 3$ input parameters.

To be able to train the ANN, a large amount of data needed to be generated. As discussed in Chapter 3, a Python program was written in preparation for this project but was accelerated using GPUs. Thus, the data samples needed to train the ANN were obtained with the GPU program from Section 3.3.2.

4.2.2 Preprocessing

The input data for the ANN in this project consisted of different values for the DM mass m_χ and the EFT coefficients c_7^s and c_7^l. The masses were generated in

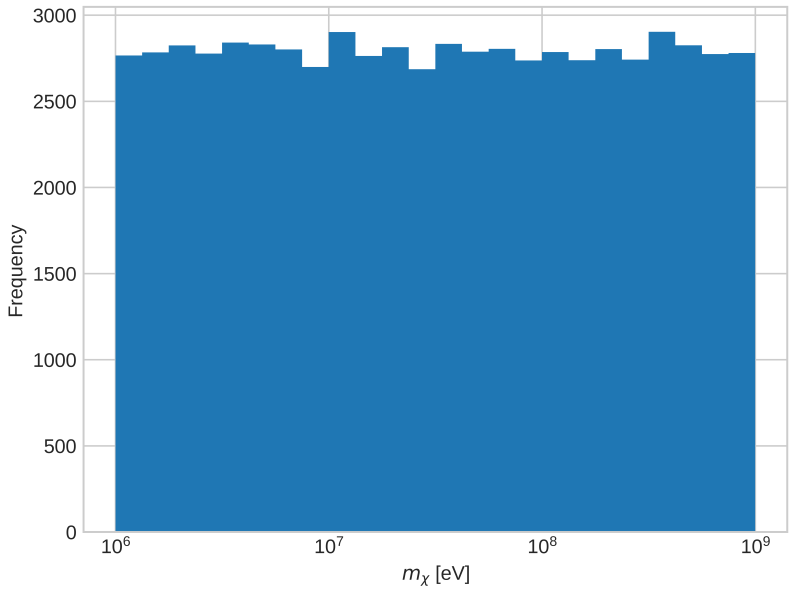


Figure 4.3: An example of an unaltered distribution of DM masses that were randomly generated with a logarithmically uniform distribution. The y-axis shows the frequency of the rates generated with a mass corresponding to the (logarithmic) x-axis.

a logarithmically uniform manner in the range of 1 MeV to 1000 MeV, while the (dimensionless) c_7^s and c_7^l coefficients were uniformly generated in the range 0 to 1. All of the other EFT coefficients were set to zero. The output data is the corresponding rates for up to $n_Q = 10$ electron-hole pairs, so the dimension of the input data is three while the output data has a dimension of ten.

As discussed in Section 4.1.3, the input data need to be preprocessed, especially the large range of logarithmically distributed DM masses. Since the transition rates follow an almost logarithmic distribution in the range 0 to 10^7 in the relevant units, it turned out to be beneficial to normalise the output data too. When building the ANN and testing the effect of different normalisation procedures, the following method performed the best: Both the input and output data were first transformed into uniform distributions between 0 and 1, and then transformed to adopt a zero mean and unit variance. Each of the features and labels was normalised individually, and the transformations were saved as objects and re-used when performing an inverse transformation. The transformations were performed with the Python package Scikit-learn [32]. Figure 4.3 shows a distribution of DM masses generated randomly with a logarithmically uniform distribution, before any transformations have been applied to the data.

The data was first transformed into a uniform distribution using the Scikit-learn class `sklearn.preprocessing.QuantileTransformer` [33]. See Figure 4.4 for an example of the transformed masses. This is a non-parametric transformation from a distribution of a feature or label generated with a random variable X and

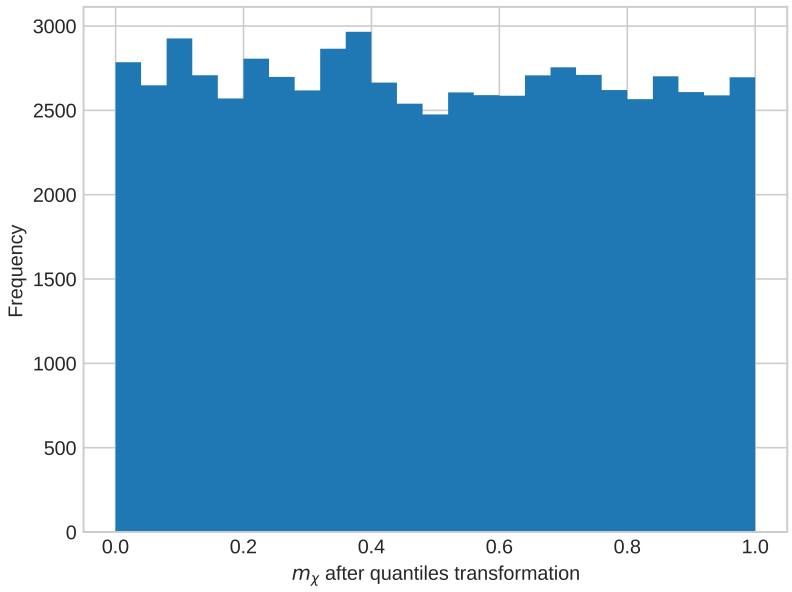


Figure 4.4: The distribution of DM masses after a quantile transformation has been applied to the distribution in Figure 4.3. The masses are now uniformly distributed in the interval $[0, 1]$.

cumulative distribution function F into a desired distribution G. The quantile function corresponding to the distribution G is then given by

$$Q_G(p) = \min \{x \in \mathbb{R} : p \leq G(x)\} . \tag{4.6}$$

The transformation $Q_G(F(X))$ then gives the desired output because of the following observations: (1) If $F(X)$ is continuous then $F(X)$ will be uniformly distributed in $[0, 1]$, and (2) if Y is a random variable with uniform distribution in $[0, 1]$ then $Q_G(Y)$ has distribution G [34]. Thus, $Q_G(F(X))$ will have the distribution G. In this project, G is a uniform distribution in the interval $[0, 1]$.

Secondly, the class `sklearn.preprocessing.StandardScaler` from Scikit-learn was used to transform the data into a uniform distribution with zero mean and unit variance [31]. See Figure 4.5. The features and labels were transformed individually from a distribution of samples x into samples z via $z = (x - \bar{x})/\sigma_x$, where $\bar{x}$ is the sample mean and σ_x is the sample standard deviation.

The original features and labels are easily recovered by applying the inverse transformations; first the inverse of the standard scaler, and then the inverse of the quantile transformer. Any data transformation procedure alters the data and leads to information losses. In this case, the quantile transformer distorts correlations and distances within and across features or labels [34]. This has not led to any apparent problems with the data used in this project, but it is important to be aware of. Without the quantile transformer, the data could not be processed by the ANN.

One of the greatest advantages of using the quantile transformer followed by the standardising transform is that they can be used on different types of data. For

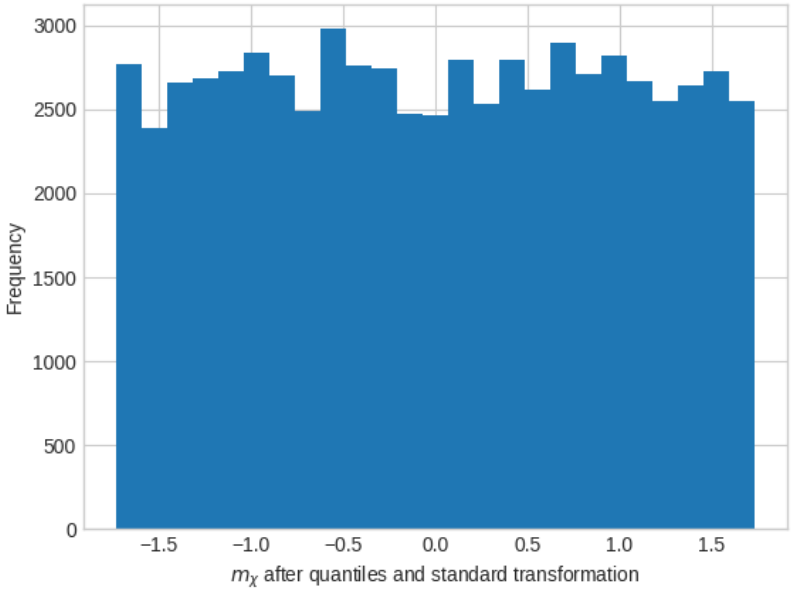


Figure 4.5: After applying a standardising transformation to the distribution in Figure 4.4, the DM masses are distributed uniformly with zero mean and unit variance. This is the type of distribution that works well with machine learning routines.

example, both the logarithmically uniform masses and the uniform EFT coupling strengths can be processed with the same types of transformations as the transition rates.

4.2.3 The layout of the neural network

The file containing the ANN layout is made up of three parts: Preprocessing the data, training the ANN, and visualising the performance of the ANN.

The data is generated in a separate file, which outputs the set of parameters corresponding to the set of transition rates for different numbers of electron-hole pairs. In this project, the set of features is $\{m_\chi, c_7^s, c_7^l\}$, while there are $n_Q = 10$ labels for each set of parameters. The data is generated for the specified number of data samples and is then loaded in the ANN file.

The features, `X`, and the labels, `Y`, are loaded in the ANN file. Before training the ANN, the data is preprocessed. First, they are split into a training set and a validation set using `sklearn.model_selection.train_test_split()` from Scikit-learn [35]. This function shuffles the feature-label pairs and assigns 67% of the initial data set into a training set $\{$`X_train`, `Y_train`$\}$ and 33% into a validation set $\{$`X_val`, `Y_val`$\}$. For the remainder of this thesis, the "data size" refers to the number of samples before splitting the data. The data set is shuffled to remove any bias that could be introduced by the generation of the data. Lastly, the normalisation procedure described in Section 4.2.2 is applied. In particular, the transformations of the features and labels are given by transforming the training sets, and these same transformations are then used on the validation set. This ensures that the

validation set does not affect the ANN in any way and that the validation set has followed the same normalisation procedure as the data that the ANN has been trained with. The latter is a requirement for using the ANN. Furthermore, when using the ANN for predictions outside of a training context, no labels are given to the ANN but the ANN will output normalised rates. This means that the normalisation transformations need to be found from a training set, and then applied as inverse transformations on the output of the ANN.

After the data has been preprocessed, the ANN can be trained. In this project, Tensorflow Keras is used to build and train a multilayer perceptron ANN [36]. In particular, the class `tensorflow.keras.model.Sequential()` is used [37]. The ANN consists of four layers: One input layer with n_p neurons, one hidden layer with 16 neurons, a second hidden layer with 32 neurons, and one output layer with n_Q neurons. The hidden layers were activated with the rectified linear unit (ReLU) activation function, defined as $g(b) = \max(0, b)$, which is a good choice for regression problems. The output layer has no activation. After the layout of the neuron layers is defined, the model is built and compiled with an Adam optimiser and MSE loss function. The Adam optimiser uses stochastic gradient descent, which is described in Section 4.1.2.

This setup proved to perform relatively well for both $n_p = 1$ and $n_p = 3$, with $n_Q = 10$. A summary is presented in Table 4.1. When the above steps have been performed, the ANN is saved and used for predictions of rates `Y` given input parameters `X`. These predictions are then compared to the validation set labels. A detailed discussion on the results of the ANN is given in Chapter 5.

Table 4.1: Summary of the setup and parameter choices for the ANN developed in this project.

ANN layers	Input layer n_p neurons Hidden layer 16 neurons (ReLU) Hidden layer 32 neurons (ReLU) Output layer n_Q neurons
Optimiser	Adam
Loss function	MSE
Learning rate	0.001
Number of epochs	30

5

Results and discussion of the implementations

In this chapter, the results of implementing the ANNs and a GPU accelerated program for speeding up the computations of electronic transition rates in direct detection experiments are presented. The first two sections cover the accuracy of the predictions, first in the simpler case of having only the DM mass as an input parameter, and then in the extended case of adding two EFT parameters. The chapter concludes with a section comparing the speed of the neural network, the GPU implementation and the original program. All results in this chapter were produced for silicon crystal detectors.

5.1 The neural network with m_χ as input

As a first step in developing the neural network, only the DM mass was an input parameter. All EFT coupling strengths were set to zero with the exception of $c_7^s = 1$. First, the training of the neural network is discussed. Then the performance is studied first qualitatively, then quantitatively.

5.1.1 The loss function

When exploring different layouts for the ANN, the MSE was an important indication of the performance; see Section 4.1.2. Therefore, the MSE was plotted for both the training and validation data as a function of the number of performed epochs for the ANN. Figure 5.1 shows the MSE during training for both the training and validation data with the layout summarised in Table 4.1, for $n_p = 1$ parameter and up to $n_Q = 10$ electron-hole pairs. This ANN was trained on 10^5 data samples.

In the figure, we see that the MSE for the training set is decreasing and flattening, as expected, but that it is also decreasing for the validation set. While the MSE for the validation set usually starts to increase as a result of overfitting in most applications, it does not for this problem. The labels have no error since they have been generated with a separate, very accurate program, which means that some degree of overfitting still keeps the ANN predictions close to the generated labels. In this project, overfitting was identified by the validation MSE fluctuating heavily. This could happen when the number of hidden layers or neurons in each hidden layer was too large, for example.

The converged value for the MSE is also visible in Figure 5.1. When the training

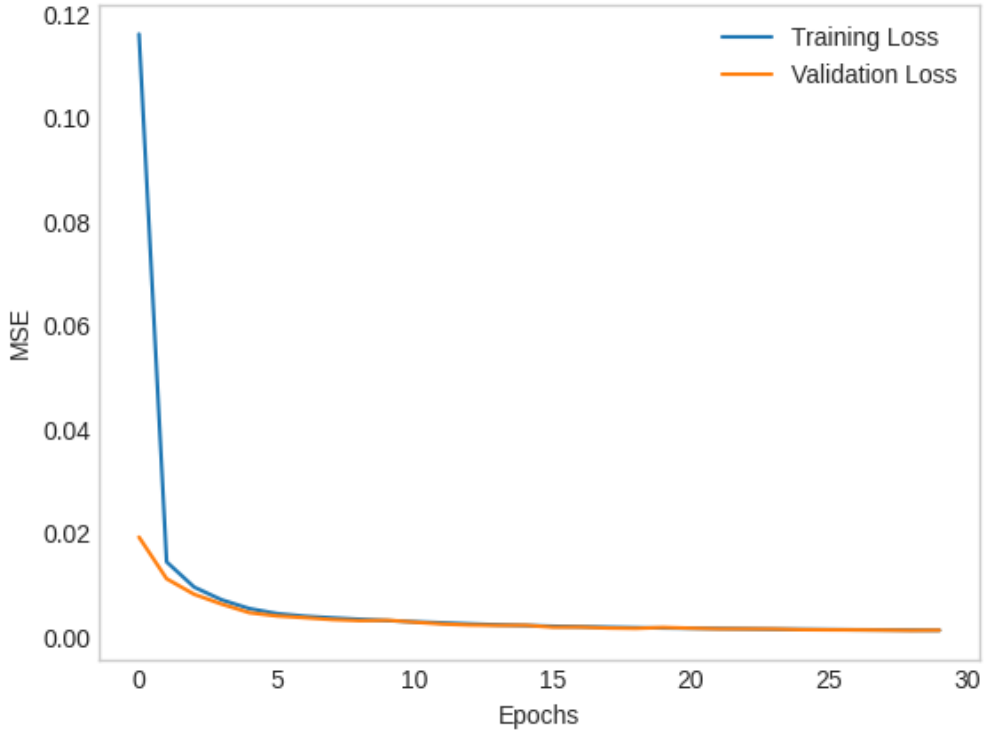


Figure 5.1: The MSE for both the training and validation data as a function of the number of performed epochs in the ANN. Both of the MSE functions decrease and flatten as the number of epochs increases. This particular figure was produced with the layout of Table 4.1 for $n_p = 1$, $n_Q = 10$, and 10^5 data samples.

and validation MSE had converged, the MSE was around 0.005. This number was deemed good enough, while higher numbers were considered as indications that the current setup was not optimal. For example, the MSE converged towards around 0.2 when the neural network was first developed.

It is also apparent that the MSE converges relatively quickly, which motivated the choice of training the ANN during just 30 epochs. It is possible to let the MSE converge slower by decreasing the learning rate for the gradient descent optimisation, but this did not improve the performance of the ANN in any noticeable way.

5.1.2 The behaviour of the neural network model

After the ANN was trained, its performance could be further assessed by visually inspecting the behaviour of the model, i.e. the predictions of the transition rates as a function of the DM mass. The DM mass was randomly generated in the interval $10^6 - 10^9$ eV with a logarithmically uniform distribution, while $c_7^s = 1$ with the rest of the EFT coupling strengths set to zero. The results in this section were produced with the setup presented in Table 4.1 with 10^5 data samples.

The advantage of having only one label was that it enabled a more intuitive in-

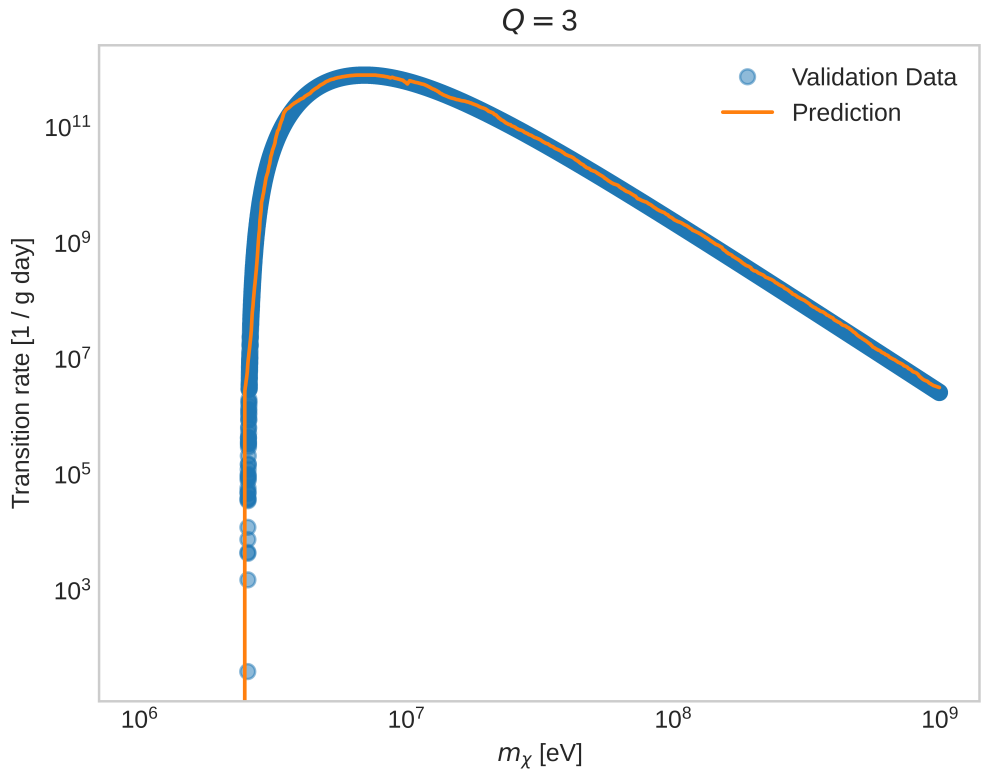


Figure 5.2: The electronic transition rate in silicon detectors as a function of the DM mass when $c_7^s = 1$ for $Q = 3$. The blue circles represent the validation data and the orange curve represents the ANN prediction. The ANN predicts the behaviour of the transition rates well over the whole range of masses.

depth analysis of the ANN. With only one label, the transition rate corresponding to a certain number of electron-hole pairs, Q, could easily be plotted against the DM mass. This proved to be a powerful tool in analysing different layer layouts in the ANN. In Figure 5.2, the transition rate as a function of m_χ in the case of $Q = 3$ is shown as an example. The blue circles represent the validation labels, while the orange curve represents the ANN prediction. In this figure, the prediction closely follows the behaviour of validation data on the whole interval. When developing the ANN, it was important that this kind of plot showed (1) that there were no large fluctuations for the prediction (a sign of overfitting), (2) that the prediction followed the behaviour of the labels at the endpoints of the interval of m_χ, and (3) that the prediction performed well around the peak of the transition rate function. These seemed to be the most problematic areas when developing the ANN. The fact that Figure 5.2 and similar figures for different values of Q fulfil these three requirements motivated the ANN setup in Table 4.1.

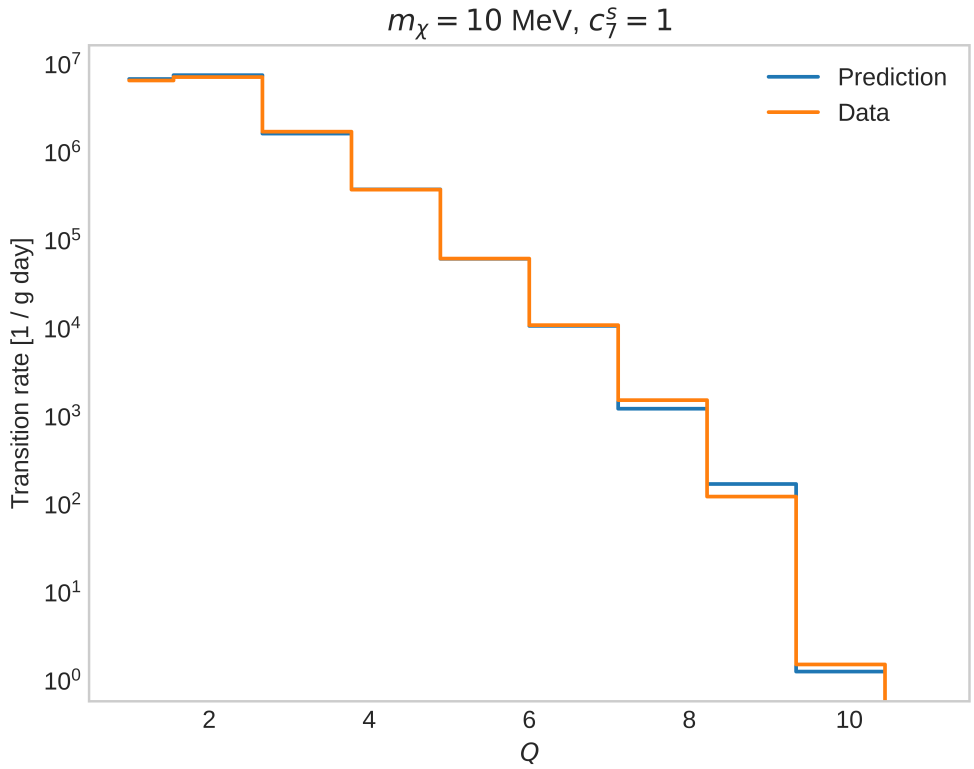


Figure 5.3: The electronic transition rate in crystal detectors as a function of the number of electron-hole pairs, Q, when $c_7^s = 1$. The generated transition rate is represented by the blue graph, while the ANN prediction is represented by the orange graph. This type of figure is further indication that the ANN performs well.

After the MSE and the transition rates as a function of m_χ had been monitored, the transition rates as a function of Q were analysed for specific values of m_χ. Figure 5.3 shows the transition rate as a function of Q with $m_\chi = 10$ MeV and $c_7^s = 1$. The blue curve shows the transition rates predicted by the ANN and the orange curve shows the generated transition rates. The figure shows that the ANN predicts the behaviour of the transition rates very well, and performs better for the lower values of Q than the higher.

5.1.3 The accuracy of the neural network

Finally, the accuracy of the predictions was quantified by the relative error for prediction i,

$$\text{error}(i) \equiv \frac{\text{prediction}(i) - \text{generated}(i)}{\text{generated}(i)}. \tag{5.1}$$

With this quantity, we can find the distribution of errors made by the ANN when it predicts the transition rate for a set of input parameters. Figure 5.4 illustrates the distribution of prediction errors when the ANN predicts the transition rates for 10^3 input samples and $Q = 3$, together with the mean, median and standard deviation

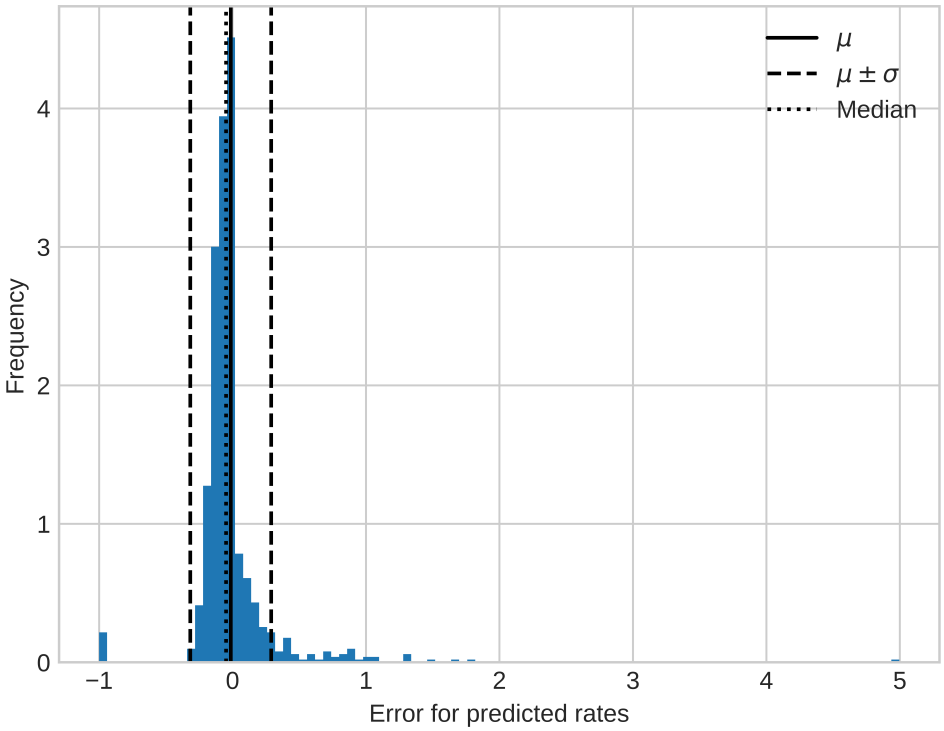


Figure 5.4: The distribution of errors, see (5.1), for 10^3 predictions made by the ANN in Table 4.1 in the case of only m_χ as an input parameter. This figure is an example of the error distribution for $Q = 3$ and both rates equalling zero and outliers (errors over 10) were removed. For $Q = 3$, there were 12 outliers out of 10^3 predictions.

of the predicted transition rates. The figure was produced by removing outliers, i.e. predictions with an error larger than 10, for illustration purposes. Since the ANN is excellent at predicting when the transition rate is zero, those predictions were removed for the sake of being able to use the error defined in (5.1). The median was also included since it is more robust to outliers and can give a better indication of the most frequent errors than the mean.

From Figure 5.4, it is observed that while the mean error is very close to zero, the standard deviation is relatively large. The error distributions for all values of Q look similar to Figure 5.4 with different values for the statistics. The main similarities are that the mean and median are usually negative, that the larger errors are usually overestimations, and that the number of outliers increases for larger values of Q. In the figure for $Q = 3$, we see that there is one occurrence out of 10^3 when the ANN produces a prediction that is off by a factor of 5. There were also 12 occurrences out of 10^3 predictions when the error was larger than 10. Furthermore, there exists a very small risk for extremely big (positive) outliers. For example, when the ANN predicts 10^6 transition rates, one of the predictions can have a positive error of the order 1000.

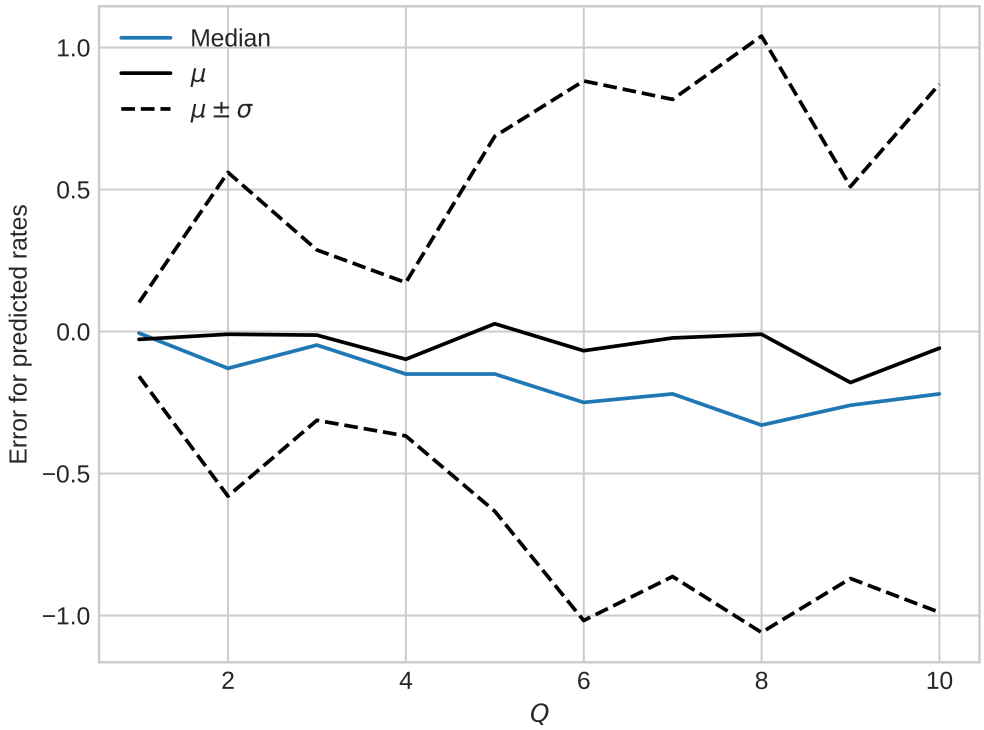


Figure 5.5: The mean, median and standard deviation of the prediction error, see (5.1), made by the ANN for different values of Q in the case of one input parameter, m_χ. The mean and median tend to be close to zero but negative, and the standard deviation seems to increase for larger values of Q.

Figure 5.5 shows the mean, median and standard deviation for the prediction errors for different values of Q. It is accompanied by Table 5.1, which includes the specific numbers for the error statistics for up to $Q = 5$. Again, errors larger than 10 and transition rates equalling zero were excluded. The graphs for the mean and median in the figure shows that the ANN tends to slightly underestimate the transition rates. It is also apparent that the standard deviation increases for larger values of Q. As previously mentioned, the number of large overestimations increases for larger values of Q, which explains why the mean stays close to zero when the median seems to decrease for larger values of Q. This completes the picture of the general behaviour of the ANN; there are more underestimations than overestimations, but the large outliers are always overestimations.

Finally, it is clear that the picture painted by Figures 5.4 and 5.5 fits well with the predictions in Figure 5.3. In Figure 5.3, the predictions are good for up to $Q = 5$, but then get increasingly worse. It is also important to note that while the standard deviation of the prediction error is relatively large for the different values of Q, an error of e.g. 0.2 makes little difference in a figure such as Figure 5.3. This highlights the fact that the ANN is very good at predicting the behaviour of the transition rates on a qualitative level.

Table 5.1: The median, mean and standard deviation of the one-input ANN prediction error for up to $Q = 5$.

Q	1	2	3	4	5
Median	-0.0060	-0.13	-0.048	-0.15	-0.15
μ	-0.028	-0.010	-0.013	-0.098	0.027
σ	0.13	0.57	0.30	0.27	0.66

5.2 The neural network with m_χ, c_7^s and c_7^l as inputs

The second version of the ANN accepts three input parameters; the DM mass, m_χ, and the EFT coupling strengths c_7^s and c_7^l. The mass was generated randomly with a logarithmically uniform distribution in the range $10^6 - 10^9$ eV as in Section 5.1, while the (dimensionless) coupling strengths were generated uniformly in the range $0 - 1$. The other coupling strengths were set to zero. The ANN layout presented in Table 4.1 was used in this section, for 10^6 data samples and now with $n_p = 3$.

To analyse the performance of the ANN for this layout, a similar procedure to the case of Section 5.1 is followed. First by getting a sense of the quality of the predictions by plotting the transition rates as a function of Q and then by quantifying the prediction error for different values of Q.

5.2.1 The behaviour of the model in the three-input case

When training the ANN for three input parameters, the first step was to study the loss function in exactly the same way as in Section 5.1.1. However, in this case, it is hard to interpret the performance of the ANN based on results such as Figure 5.2, since the transition rates now vary with three different parameters. Instead, with the knowledge that the layout in Table 4.1 worked well in the case of one input parameter, the three-input ANN was based on the network developed for the one-input ANN.

The top figure of Figure 5.6 is the equivalent of Figure 5.3 in the case of three input parameters. For the more complex case of including two EFT parameters, the prediction made by the ANN is relatively good. It does not appear as accurate as in the one-input case, but it captures the overall behaviour of the electronic transition rates well. Furthermore, it seems to be better at predicting the electronic transition rates for larger values of Q.

The middle figure of Figure 5.6 illustrates the transition rates predicted by the ANN for long-range interactions, when $c_7^s = 0$ and $c_7^l = 1$ for $m_\chi = 10$ MeV. This is a significant improvement from the case of just one input parameter; while the one-input model does seem more accurate, the three-input ANN has learned to distinguish between the effects of short- and long-range interactions. Even though

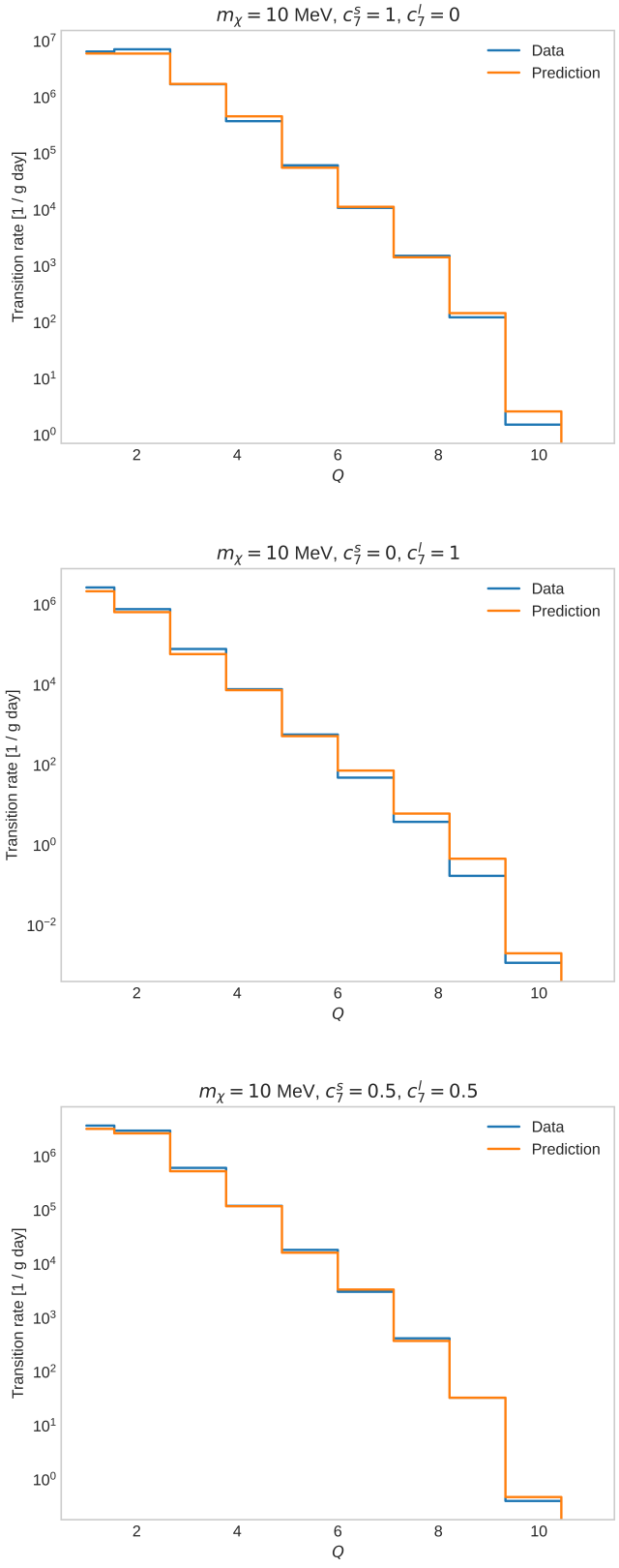


Figure 5.6: The ANN prediction of electronic transition rate as a function of the number of electron-hole pairs, Q, and the directly generated transition rates for different combinations of the three input parameters. All three show $m_\chi = 10$ MeV. Top shows $c_7^s = 1$, $c_7^l = 0$; middle $c_7^s = 0$, $c_7^l = 1$; bottom $c_7^s = 0.5$, $c_7^l = 0.5$.

the order of the transition rates for the short- or long-range interactions vary dramatically between the top two figures of 5.6, the bottom figure shows that the ANN can even predict transition rates based on a mix of c_7^s and c_7^l.

5.2.2 The accuracy of the predictions

As in the case of the one-input model in Section 5.1, (5.1) is used to quantify the error. Figure 5.7 shows the distribution of the prediction errors made by the three-input ANN when predicting the transition rates for 10^3 data samples. As in the one-input case, predictions equalling zero as well as errors over 10 were removed when analysing the distribution of errors. In total, the ANN usually produces less than 10 outliers when predicting 10^3 transition rates for all values of Q. This is about the same frequency as in the one-input case. In Figure 5.7, there were no outliers.

Since Figure 5.6 only shows single predictions made by the ANN, Figure 5.7 complements it by showing the full spectrum of errors. It shows that the ANN seems to systematically underestimate the electronic transition rates for $Q = 0$, which is slightly visible in Figure 5.6. Furthermore, the distribution of errors is not completely contained around the mean; there are many occurrences where the

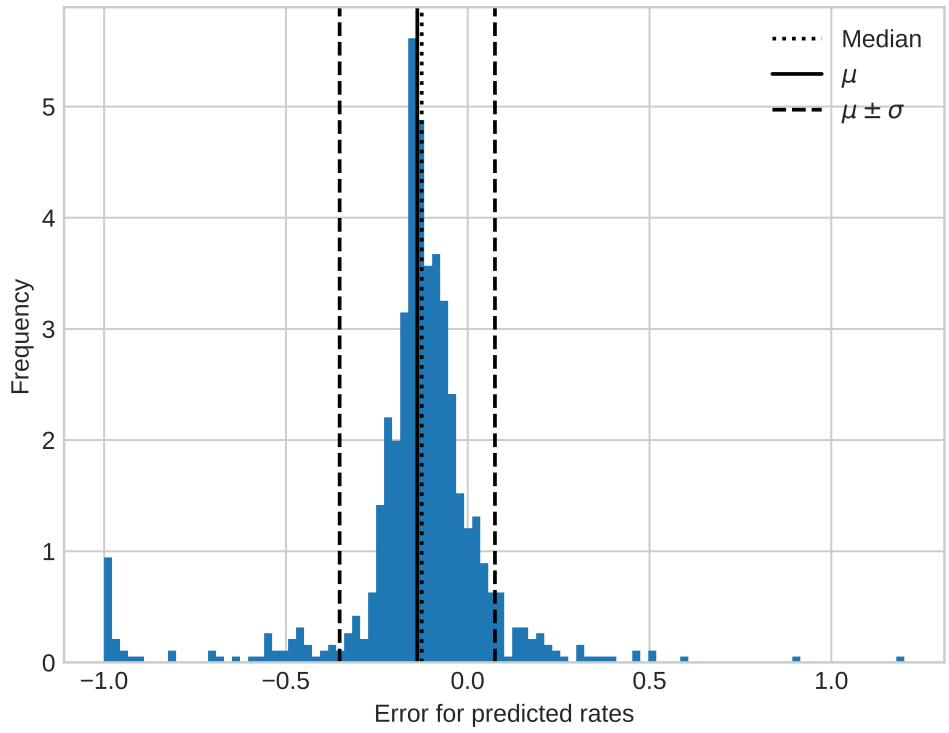


Figure 5.7: The distribution of errors, defined in (5.1), for the predictions made by the ANN when $Q = 1$. This ANN was trained on 10^6 data samples, and 10^3 transition rates were predicted to generate this figure.

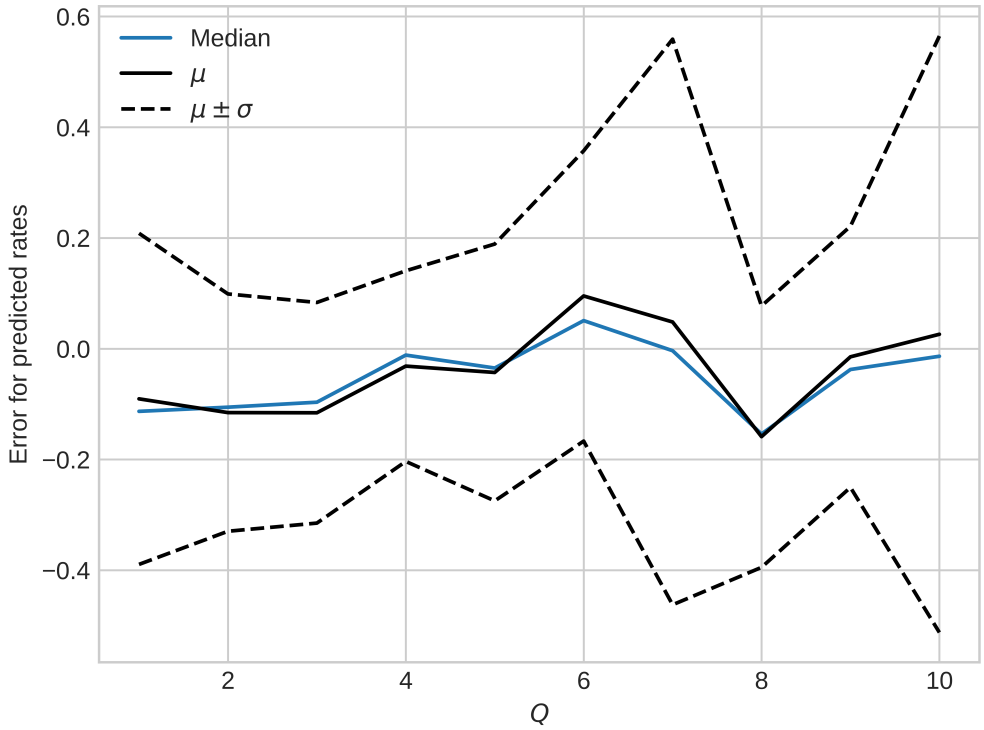


Figure 5.8: The prediction error, see (5.1), for different Q made by the ANN trained on 10^6 data samples. The errors were produced for a data set containing 10^3 data samples. The outliers, i.e. errors larger than 10, and rates equalling zero were removed from the data set.

predictions are far from zero.

In Figure 5.8, the mean, standard deviation and median of the distribution of prediction errors are shown for different values of Q. It is not as clear as in Figure 5.5 that the standard deviation of the errors increases for larger values of Q, as was suspected from Figure 5.6. There also seems to be a larger offset from a zero mean, further indicating that the three-input ANN is not as accurate as the one-input ANN.

Finally, Table 5.2 shows the means, standard deviations and medians correspond-

Table 5.2: The median, mean and standard deviation of the three-input ANN prediction error for up to $Q = 5$.

Q	1	2	3	4	5
Median	−0.11	−0.11	−0.097	−0.012	−0.035
μ	−0.091	−0.12	−0.12	−0.031	−0.043
σ	0.30	0.21	0.20	0.17	0.23

ing to Figure 5.8. This can be compared to Table 5.1, where it becomes apparent from the mean and median that the three-input ANN systematically underestimates the transition rates to a higher degree, at least for the first five values of Q. On the other hand, the difference between the standard deviations is less than expected.

5.3 Summary and comparison of improvements

Finally, the speed of all implementations discussed in this thesis can be compared. The speed was found by measuring the execution time for each program as a function of the number of computed (or predicted) transition rates. In Figure 5.9, the speed of the Python program (Section 3.1), the C++ program running on the CPU (Section 3.2), the C++ program running on the GPU (Section 3.3.2) and the three-input ANN (Section 4.2.3) are presented. The ANN used the GPU program for generating the training data. For this figure, all four implementations computed the electronic transition rates corresponding to different values for m_χ, c_7^s and c_7^l when the other EFT coupling strengths were set to zero. The speed results were obtained on a computer with an AMD Ryzen 7 3750H CPU and an Nvidia GeForce GTX 1660 Ti GPU.

As expected, for larger numbers of computed transition rates the original Python

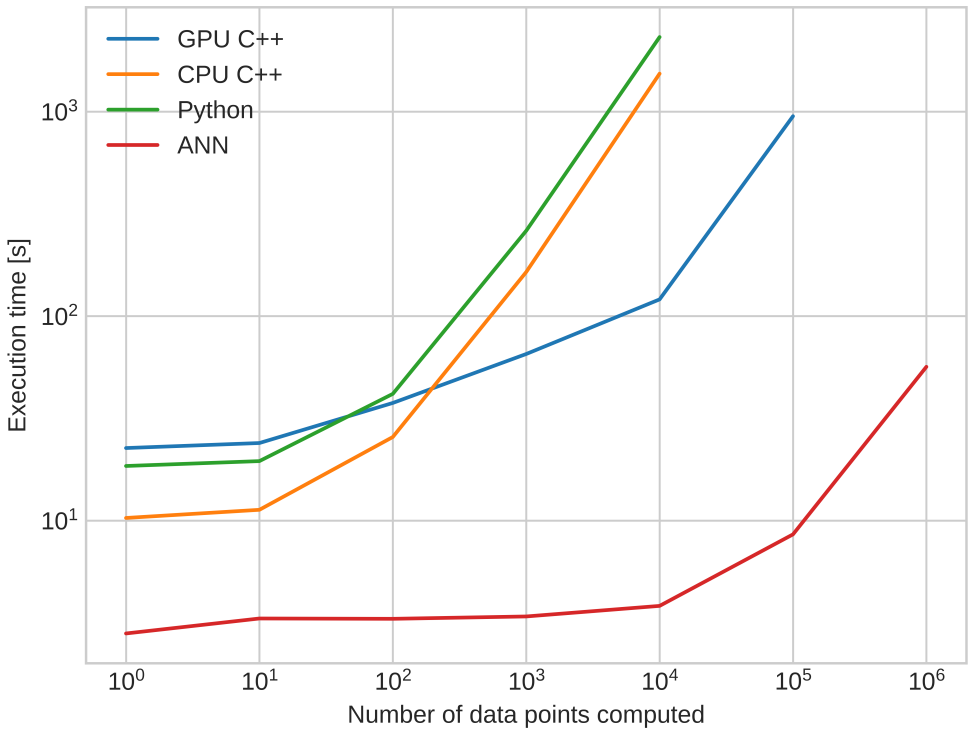


Figure 5.9: A speed comparison of the Python program, the C++ CPU program, the GPU program and the ANN predictions.

program is the slowest implementation, followed by the C++ CPU program which is about twice as fast as the Python program. The GPU program is the slowest implementation for a low number of computed transition rates but eventually becomes much faster than the Python program. This is because of the memory transfer overhead that was discussed in Section 3.3.1, but the large amount of available threads makes the GPU program very powerful for a larger amount of computed transition rates.

In Table 5.3, the execution times used to generate Figure 5.9 are presented. Some time measurements were left out for the C++ CPU and Python programs since they are very computationally costly. Now, it becomes clear that the GPU program is about 16 times faster than the Python program, and that the ANN is about 600 times faster than the Python program for 10^4 computed transition rates. For 10^6 computed transition rates, the ANN is about 160 times faster than the GPU program. Note that the time for training the ANN is not included, but since it is a one-time computation and it is less than one hour, it is not relevant for the speed comparisons. The results are further discussed in Chapter 5.

Table 5.3: The execution times for the four implementations as a function of the number of transition rates computed.

n_D	1	10	100	1000	10^4	10^5	10^6
ANN	3s	3s	3s	3s	4s	9s	56s
GPU	23s	24s	38s	1min5s	2min1s	15min49s	2h30min
CPU	10s	11s	25s	2min44s	25min33s	–	–
Python	18s	19s	41s	4min21s	38min34s	–	–

6
Discussion and outlook

In this chapter, the results obtained in Chapter 5 are discussed in detail. First, improvements are suggested both for the GPU and ANN implementations. Then possible expansions of the thesis are presented, followed by the final conclusions of the results in this thesis.

6.1 Improving the results

The full details of the results achieved in this thesis are presented in Chapter 5. In this section, the results are further discussed and possible improvements are suggested.

6.1.1 The performance of the GPU program

The GPU program was developed with the purpose of speeding up the data generation for the ANN. However, it turned out that the GPU program could compete with the ANN both in speed and accuracy. The GPU program does not introduce any error in comparison to the Python program, meaning that the error statistic defined in (5.1) is zero for all data samples generated by the GPU. However, the ANN is currently significantly faster than the GPU program.

As mentioned in Section 3.3.2, only one-dimensional threads were used in the GPU program. If more dimensions were to be used, there is great potential for the GPU to be even faster. With the added accuracy of the GPU program, GPUs are a strong contender for the fastest method of computing electronic transition rates. Furthermore, the speed results presented in this thesis were obtained on a personal laptop GPU. If the code were to be optimised for a powerful high-performance GPU instead, the results could improve significantly.

6.1.2 The performance of the neural network

While the final ANNs predicted the overall behaviour of the electronic transition rates as a function of both one and three parameters very well, it was challenging to achieve high accuracy with the setup in Table 4.1. When going from the one-input ANN to the three-input ANN, it was not obvious that the two networks should have the same setup in regard to the number of hidden layers and the number of neurons in each hidden layer. Therefore, many different setups were explored; examples include up to five hidden layers, up to a hundred neurons in the hidden layers, different ratios

of neurons between the hidden layers and different regularisation schemes. These options were investigated with the aim of achieving higher accuracy for the ANN while studying the MSE during training and the behaviour of the ANN such as in Figure 5.6. These attempts usually led to overfitting, or in some cases a good model, but never as good as for the setup in Table 4.1. This is probably because the setup of one input neuron, 16 in the first hidden layer, 32 in the second and 10 neurons in the output layer is not that different from having three input neurons. In other words, the setup that worked best for the one-input ANN is probably the one that is also best for the three-input ANN. This might not be the case if there were e.g. 10 input neurons, however.

The final version of the one-input ANN was developed by letting the ANN train on 10^5 data samples. Before choosing this size of the data samples, the performance based on 10^4 data samples was investigated. In that case, the result corresponding to Figure 5.3 showed that the ANN was somewhat good at predicting the transition rates, but not as well as when the ANN had trained on 10^5 data samples. Therefore, the ANN was also trained on 10^6 data samples to investigate if a larger training data size could improve the results further. This did not lead to any apparent improvement in the accuracy; instead, the ANN performed as well as when it trained on 10^5 data samples. For the same reasons, the three-input ANN was also trained on 10^7 data samples, in contrast to the final ANN that was trained on 10^6 data samples. Again, this did not lead to any significant improvement in the ANN predictions.

Even though the ANN layouts are very similar, the one-input ANN performs slightly better than the three-input ANN. This is most likely because the introduction of two EFT parameters drastically increases the complexity of the transition rate model. While the current layout for the ANN does not seem to be possible to further improve, there are still many aspects that can be investigated to increase the accuracy of the ANN. Most importantly, only different layouts of a multilayer perceptron ANN have been investigated in this thesis. Naturally, the next step in improving the accuracy of the ANN is to implement different architectures. This becomes especially relevant when the dimension of the input parameter space increases.

Lastly, the ANN makes very fast predictions, as is presented in Section 5.3. While there was no significant difference in the prediction speed between the one-input and three-input ANNs, it is possible that the predictions become slower when more input parameters are added. Since the current ANN is about 160 times faster than the GPU program, however, the prediction speed for a larger number of input parameters is not concerning. The challenging aspect could instead be the time for training the ANN. The training time has not been discussed in this thesis since it has always been negligible; always under ten minutes for both ANNs, and it is only performed once before a model is produced that can predict transition rates. However, the training time increases with the training data size. Since the data size had to be increased when going from one input parameter to three, it is strongly suspected that more training data is needed when more parameters are added as inputs. This makes the training time relevant. There are ways to adjust the training of the ANN developed in this thesis, such as changing the batch size, but it would also be interesting to implement an ANN that is trained on the GPU.

6.2 Expanding the project

With this thesis, the advantages of using GPUs and ANNs to accelerate computations of electronic transition rates have been demonstrated. Furthermore, there is a high potential for improvements in accuracy.

To simplify the analysis of the ANN, a one-input and then a three-input ANN were developed. However, a total of 28 EFT coupling strengths were presented in Section 2.1.4. To fully describe the DM-electron scattering process with this EFT, all coupling strengths should be included as input parameters. A reasonable first step could be to include the $\mathcal{O}_8$ interactions since it is another simple operator containing both spin and velocity dependence, other than $\mathcal{O}_7$. Since the complexity of the model increases significantly with every added parameter, I believe the best procedure consists of adding a few input parameters at a time and evaluating the performance of the new ANN; this was the procedure used in this thesis, and studying the one-input ANN significantly facilitated the development of the three-input ANN. This is probably also the case for developing ANNs with other architectures.

The reason for generating the c_7 coupling strengths uniformly in the range $0-1$ was that since they originate from the same operator, $\mathcal{O}_7$, any relative difference between c_7^s and c_7^l larger than a few orders of magnitude would lead to one coupling strength clearly dominating over the other, as is seen in (2.45). If more EFT coupling strengths would be added as inputs to the ANN, they might need to be generated logarithmically to account for differences in magnitude between different operators.

Furthermore, parameters other than the coupling strengths can be added as inputs to the ANN. For example, the velocities v_0, $v_\oplus$ and v_{esc} in Section 2.1.2 could be further probed by using them as input parameters for the ANN.

A natural extension of this project would be to apply the ANN for interpretations of direct detection experiments. The reason for computing the electronic transition rates was to obtain a likelihood function describing the distribution of the transition rates depending on different sets of input parameters, which can then be used for parameter estimation via Bayes' theorem. Now that there exists a GPU parallelised program as well as a ANN for computing the electronic transition rates, the performance for evaluating the likelihood function can be analysed.

Finally, this thesis has demonstrated two methods of significantly accelerating computations for DM-electron scattering in crystal detectors. Since these methods are not specific to the physical phenomenon studied in this thesis, they can be applied to other research areas that depend on fast computations. In particular, the results of this thesis have shown that both GPU computations and ANN predictions are fast, viable options for modelling the behaviour of an EFT-based theory.

6.3 Conclusions

With this thesis, two approaches have been implemented for accelerating the computation of electronic transition rates in crystal direct detection experiments. The use of both GPUs and ANNs for scientific purposes is relatively new, and has been applied here for the purpose of advancing dark matter research.

The ANN was able to learn the behaviour of the electronic transition rates as a function of the DM mass and two EFT coupling strengths. This implementation provided a speed-up of a factor of 600 compared to the program used prior to this thesis. The ANN still makes prediction errors but could be well-suited for e.g. distinguishing transition rates corresponding to different sets of input parameters. With further improvements, the ANN could become even more accurate and eventually provide a very fast, highly accurate model for the electronic transition rates.

The GPU program proved to be a strong competitor to the ANN. It is about 16 times faster than the previous implementation for computing the electronic transition rates with the same accuracy as the previously used program. This means that the GPU program is both fast and highly accurate, and there are several suggestions in this thesis for how the GPU computations could be further accelerated. Furthermore, the speed of the GPU program does not depend on the number of non-zero EFT coupling strengths; the program developed in this thesis transfers all 28 coupling strengths to the GPU regardless of whether they are initialised to zero or randomly generated. This means that the GPU program gives a speed-up of a factor of 16 that computes the electronic transition states described by the whole EFT. Thus, the GPU program is fully prepared to be used for parameter estimation using direct detection experimental data.

Bibliography

[1] G. Bertone, D. Hooper, and J. Silk, "Particle dark matter: evidence, candidates and constraints," *Physics Reports*, vol. 405, no. 5-6, pp. 279–390, jan 2005. [Online]. Available: https://doi.org/10.1016%2Fj.physrep.2004.08.031

[2] P. Côté, M. Mateo, E. W. Olszewski, and K. H. Cook, "Internal kinematics of the andromeda ii dwarf spheroidal galaxy," *The Astrophysical Journal*, vol. 526, no. 1, p. 147, nov 1999. [Online]. Available: https://dx.doi.org/10.1086/307999

[3] F. Zwicky, "Republication of: The redshift of extragalactic nebulae," *General Relativity and Gravitation*, vol. 41, no. 1, pp. 207–224, Jan. 2009.

[4] W. Hu and S. Dodelson, "Cosmic microwave background anisotropies," *Annual Review of Astronomy and Astrophysics*, vol. 40, no. 1, pp. 171–216, sep 2002. [Online]. Available: https://doi.org/10.1146%2Fannurev.astro.40.060401.093926

[5] G. Bertone and D. Hooper, "History of dark matter," *Reviews of Modern Physics*, vol. 90, no. 4, oct 2018. [Online]. Available: https://doi.org/10.1103%2Frevmodphys.90.045002

[6] T. M. Undagoitia and L. Rauch, "Dark matter direct-detection experiments," *Journal of Physics G: Nuclear and Particle Physics*, vol. 43, no. 1, p. 013001, dec 2015. [Online]. Available: https://doi.org/10.1088%2F0954-3899%2F43%2F1%2F013001

[7] CERN. (2023) High luminosity lhc project. [Online]. Available: https://hilumilhc.web.cern.ch/

[8] R. Catena, T. Emken, M. Matas, N. A. Spaldin, and E. Urdshals, "Crystal responses to general dark matter-electron interactions," *Physical Review Research*, vol. 3, no. 3, aug 2021. [Online]. Available: https://doi.org/10.1103%2Fphysrevresearch.3.033149

[9] A. L. Fitzpatrick, W. Haxton, E. Katz, N. Lubbers, and Y. Xu, "The effective field theory of dark matter direct detection," *Journal of Cosmology and Astroparticle Physics*, vol. 2013, no. 02, pp. 004–004, feb 2013. [Online]. Available: https://doi.org/10.1088%2F1475-7516%2F2013%2F02%2F004

[10] R. Catena, T. Emken, N. A. Spaldin, and W. Tarantino, "Atomic responses to general dark matter-electron interactions," *Physical Review Research*,

vol. 2, no. 3, aug 2020. [Online]. Available: https://doi.org/10.1103%2Fphysrevresearch.2.033195

[11] M. E. Peskin and D. V. Schroeder, *An introduction to quantum field theory*. Boulder, CO: Westview, 1995.

[12] R. Essig, M. Fernandez-Serra, J. Mardon, A. Soto, T. Volansky, and T.-T. Yu, "Direct detection of sub-gev dark matter with semiconductor targets," 2015. [Online]. Available: https://arxiv.org/abs/1509.01598

[13] F. Mandl and G. Shaw, *Quantum field theory*, 2nd ed. Hoboken, N.J: Wiley, 2010, oCLC: ocn460050759.

[14] J. J. Sakurai and J. Napolitano, *Modern quantum mechanics*, 3rd ed. Cambridge: Cambridge University Press, 2021.

[15] J. Fan, M. Reece, and L.-T. Wang, "Non-relativistic effective theory of dark matter direct detection," *Journal of Cosmology and Astroparticle Physics*, vol. 2010, no. 11, pp. 042–042, nov 2010. [Online]. Available: https://doi.org/10.1088%2F1475-7516%2F2010%2F11%2F042

[16] P. Hofmann, *Solid state physics: An introduction*, 2nd ed. Weinheim, Germany: Wiley-VCH, 2015.

[17] G. B. Folland, *Fourier analysis and its applications*. Rhode Island, USA: American Mathematical Society, 2009.

[18] L. Råde and B. Westergren, *Mathematics handbook for science and engineering*, 5th ed. Lund, Sweden: Studentlitteratur, 2004.

[19] C. R. H. et al, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020. [Online]. Available: https://doi.org/10.1038/s41586-020-2649-2

[20] IBM. (2010) Compiled versus interpreted languages. [Online]. Available: https://www.ibm.com/docs/en/zos-basic-skills?topic=zos-compiled-versus-interpreted-languages

[21] I. Turner-Trauring. (2023) Massive memory overhead: Numbers in python and how numpy helps. [Online]. Available: https://pythonspeed.com/articles/python-integers-memory/

[22] S. K. Lam, A. Pitrou, and S. Seibert, "Numba: A llvm-based python jit compiler," in *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, ser. LLVM '15. New York, NY, USA: Association for Computing Machinery, 2015. [Online]. Available: https://doi.org/10.1145/2833157.2833162

[23] T. M. G. of Michigan State University. (2017) Numba versus c++. [Online]. Available: https://murillogroupmsu.com/numba-versus-c/

[24] G. GCC. (2023) Options that control optimization. [Online]. Available: https://gcc.gnu.org/onlinedocs/gcc/Optimize-Options.html

[25] I. Evgueny Khartchenko. (2023) Vectorization: A key tool to improve performance on modern cpus. [Online]. Available: https://www.intel.com/content/www/us/en/developer/articles/technical/vectorization-a-key-tool-to-improve-performance-on-modern-cpus.html

[26] Nvidia, *CUDA C++ Programming Guide*, 2021, https://docs.nvidia.com/cuda/archive/11.2.0/pdf/CUDA_C_Programming_Guide.pdf.

[27] K. Group. (2023) Opencl. [Online]. Available: https://www.khronos.org/opencl/

[28] H. O. et al, "Snabba beräkningar av elastisk proton-neutronspridning med en grafikprocessor," 2021. [Online]. Available: https://odr.chalmers.se/items/bdeec4ba-81f1-47d5-b22d-ca34ffc08d49

[29] B. Mehlig, *Machine learning with neural networks: An introduction for scientists and engineers*, 1st ed. Cambridge ; New York, NY: Cambridge University Press, 2022.

[30] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[31] Scikit-learn. (2023) Standardscaler class definition. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

[32] F. e. a. Pedregosa, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[33] Scikit-learn. (2023) Quantilestransformer class definition. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.QuantileTransformer.html

[34] ——. (2023) Preprocessing data with a non-linear transformation. [Online]. Available: https://scikit-learn.org/stable/modules/preprocessing.html#preprocessing-transformer

[35] ——. (2023) Data splitting function definition. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html

[36] Keras. (2023) Keras: A tensorflow api. [Online]. Available: https://keras.io/

[37] ——. (2023) Keras sequential slass. [Online]. Available: https://keras.io/api/models/sequential/

DEPARTMENT OF SOME SUBJECT OR TECHNOLOGY
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY