Fast Mie calculations with a radial basis function neural network

Michael Jonas Stremme

Master of Science in Physics
at the department of Physics and Technology
University of Bergen, Norway

2019

Thesis date: Feb 1, 2019

Year: 2019
Title: Fast Mie calculations with a radial basis function neural network
Author: Michael Jonas Stremme

Abstract

Scattering phenomena have been of great interest and driving scientific advancement for centuries. Mie theory outlines a precise way to compute the scattering by spherical particles. However, the computational labor involved in Mie calculations are very time consuming. Computer modeling of light intensities often are heavily dependent on calculations of absorption and scattering of radiation interactions. Maxwell's equations for such large particles require very many expansion terms which slow down the computation time. Using Mie theory, this thesis aims to investigate the use of artificial neural network in order to decrease computation time for a wide variety of particle size distributions and refractive indices. The input and output parameters of the Mie code will be used to train an artificial neural network in an attempt to increase the speed of such calculations, while still being reasonably accurate.

The a single radial basis function neural network with 300-500 neurons is enough to approximate the scattering coefficient, absorption coefficient and asymmetry parameter with an mean absolute percentage error of less than 2%. However the potential of outlier errors of up to 20-30% exists with the currently iteration of neural network. The time save of a neural network can be enormous with a potential improvement of 10000 times faster than traditional Mie codes.

Acknowledgements

This master thesis process has been long but also rewarding. The topic was selected after talks with my supervisor Børge Hamre, and the combination of using machine learning algorithms to solve complex physical calculations intrigued me. Along the way I have received much support, encouragement, and help from family and friends. For this I am very grateful. Last I would like to thank my supervisor for guidance and insightful discussions along the way. Thank you.

Contents

List of Figures

List of Tables

Acronyms

ANN Artificial Neural Network. 1, 11, 12, 14

FFNN Feed Forward Neural Network. 14, 15, 19, 37

IOP Inherent Optical Properties. 4, 20, 39

MAPE mean absolute percentage error. 24, 25, 27–29, 32, 33

MAXAPE maximum absolute percentage error. 24, 25, 27–29, 32, 33

MLP Mulitlayer Perception. 14–16

MSE mean square error. 22, 25, 38

NN Neural Network. 14, 23, 25, 28, 38

PE percentage error. 24, 33

RBF Radial Basis Function. 13–16, 22, 23, 25, 33, 37–39

RBFNN Radial Basis Neural Network. 15–17, 19, 20, 22–25, 27–29, 32–34, 37–39

RNN Recurrent Neural Network. 14

SD standard deviation. 24, 25, 27–29, 32, 33

Chapter 1

Introduction

Scattering phenomena have been of great importance, sparking curiosity and driving scientific advancement. One such phenomena is the rainbow which has fascinated people for centuries and given rise to a large variety of theories. Attempts at giving a rational explanation for the phenomena date back to Aristotle. He proposed that it was due to some unusual kinda of reflection of sunlight from clouds (Nussenzveig, 1977, p.116). It wasn't until 1304 when Theodoric of Freiberg and later René Descartes independently proposed that the rainbow was due to light rays entering and reflected by the inside of individual raindrops (Nussenzveig, 1977, p.116).

The reflection and refraction of light by the water droplets was able to explain some of the main features of the rainbow, however many others still remained a mystery. The colors of the rainbow could finally be explained when Isaac Newton in 1666 demonstrated the dispersion effect by splitting white light into its composite colors in his prism experiment. The continuing advances made in the theory of light kept giving insights into more features of the rainbow. The electromagnetic theory of light, which was later proposed by James Clerk Maxwell, made it possible to give a full mathematical formulation of the problem. Using Maxwell's theory Gustav Mie outlined a precise way to compute the scattering of a plane wave of spherical particles (Mie, 1908).

Atmospheric phenomena such as the rainbow are only one of Mie theory's many areas of use. It plays an important role in radiative transfer models which are used in fields such as climate research, remote sensing, and medical diagnostics among others.

Even though the mathematical solution to the problem has been known for over a hundred year it was not until the middle of the last century that it started to see a lot of use. This was mainly due to the computation labor involved in evaluating functions such as Riccati-Bessel functions (Wriedt, 2012, p.56). With the rapid improvement of computer hardware as well as increasingly efficient algorithms Mie theory gained a lot of popularity. However computer programs for modeling the light intensities are often heavily dependent on calculation of absorption and scattering of radiation interacting with particles that have sizes up to 1000 times the radiation wavelength. Solutions of Maxwell's equations for such large particles require very many expansion terms, and these solutions have to be found for many different particle types, wavelengths, and size distributions, which consequently slow down radiative transfer models.

In recent years artificial neural networks ANN have grown in popularity and as a result have seen a number of new areas of application. As Artificial Neural Network are able to calculate large amounts of data, they could possibly significantly decrease time spent on Mie-

calculations, as suggested by (Lang, 2013). This project aims to investigate this possibility. I will look at scattering by spheres, and calculate parameters such as scattering efficiency, absorption and asymmetry factor. This will be done using a c++ code for Mie scattering developed in the Optics and atomic physics group at the department of physics and technology at the University of Bergen. Calculations will be done for a wide variety of particle size distributions and refractive indices. The input and output parameters of the Mie code will be used to train an artificial neural network in an attempt to greatly increase the speed of such calculations, while still being reasonably accurate.

Chapter 2

Light Scattering

In nature light can be observed in two ways, directly from the source or indirectly due to interaction with particles in its path. By far the most common way to observe light is indirectly, which is due to scattering. Scattering can be broadly defined as the redirection of radiation, such as light, usually due to interactions with molecules and particles (Lelli, 2014). Scattering processes are often divided into elastic and inelastic scattering. For elastic scattering the frequency of the scattered light is the same as the incident light, while the frequencies differ for inelastic scattering. In most cases elastic scattering is the dominant process, such as for energies in the visible range. Due to the scope of this thesis I will solely be considering the elastic scattering processes. Scattering processes are often accompanied by absorption, and both processes remove energy from the incident light beam. However, while scattering does this by redirecting part of the energy into other directions absorption turns the energy into other forms such as thermal energy. Extinction/attenuation is the sum of scattering an absorption.

$$extinction = scattering + absorption \tag{2.1}$$

There are a number of different theories describing lights interaction with matter. The physical interpretation of how light interacts with matter can thus vary wildly depending on which theory it is based on. In this thesis I will consider lights interaction with matter according to the electromagnetic theory in which light is a propagating wave of electric and magnetic fields.

The underlying physics of the scattering of electromagnetic waves is the same for any matter. Matter that interacts with EM-waves is known to be composed of discrete electric charges, such as electrons and protons for ordinary matter. These charges can be excited to oscillate by an external EM-field which in turn will radiate EM-waves. The superposition of secondary waves emitted from the charges in the scattering medium and the incident wave, is what is observed (Bohren and Clothiaux, 2006, ch.3.1).

Another aspect to be considered is the geometry of the particles. When light interacts with particles the scattered and absorbed light depends on the properties of the particles. For known properties of the particles such as shape, size and refractive index, the absorption can be calculated alongside a variety of properties of the scattered light such as polarization, angular distribution and intensity.

There are many theories and models that describe the elastic interaction between light and particles. These theories can be categorized into three group: rigorous theories, approximate

models and numerical methods (Jiang et al., 2012). Rigorous theories, such as Mie theory, are based on the exact solutions of Maxwell's equations. The scattering of electromagnetic waves on spherical particles can thus be analytically solved in the form of an infinite series. These solutions to Maxwell's equations are sometimes called Mie Solutions. However, these exact solutions are only known for a small variety of shapes with strong symmetries such as spheres, ellipsoids and cylinders. Approximate models on the other hand are often flexible to shape. However, these models may be restricted in other ways, for example regarding the size of the particle. The Rayleigh and geometrical optics approximations are for example only valid for wavelengths much smaller ($\lambda << r$) and larger ($\lambda >> r$) than the size of the particles respectively. In situations where neither exact nor approximate models are possible to apply, numerical methods can be used to find a solution. Although Mie theories restrictions to shape, it is widely used because many particles can be modeled to good approximation as spheres or spheroids. Sphere-like structures are frequent in nature, such that assuming scattering like a sphere can be a good first approximation. Additionally particles are often assumed to be randomly oriented such that the accumulative scattering effect of many such particles can also resemble scattering by spheres. Due to symmetry, scattering by spheres is much simpler than more complicated structures. However even for spherical particles light scattering calculations often take up a large bulk of the total computation time, such as when solving radiative transfer problems. Therefor it is no surprise that many researchers are assuming spherical particles when measuring scattering effects.

2.1 Inherent Optical properties

The way light interacts with a particle or collection of particles is described by the Inherent Optical Properties (IOP) of the particles in question. IOPs are properties of the medium and do not depend on the light field. In the following subsections follow a short summary of these properties and how they can be defined.

2.1.1 Cross sections and coefficients

Consider a beam incident on a particle, the scattering and absorption properties of the particle will change the characteristics of the beam after it passes the particle. For a monochromatic, monodirectional light beam with electromagnetic power W_0 incident on a homogeneous particle, the power in the same direction after the particle can be measured to be W_1, where $W_0 > W_1$. This difference represents the energy being removed out of the direction of the beam, we say that the particle has resulted in extinction of the beam. The rate at which the energy is removed from the beam is proportional to the incident irradiance F (Bohren and Clothiaux, 2006)

$$W_0 - W_1 = W_{ext} = C_{ext}F. \tag{2.2}$$

The proportionality constant C_{ext} is the extinction cross section, which may be thought of as the effective area of the particle contributing to the removal of energy out of the beam. This attenuation must be either due to absorption or scattering.

$$W_{ext} = W_{abs} + W_{sca} \tag{2.3}$$

Hence it useful to also define the absorption and scattering cross sections such that the power removed due to each is

$$W_{abs} = C_{abs}F, \tag{2.4}$$

and

$$W_{sca} = C_{sca}F, \tag{2.5}$$

so that

$$C_{ext} = C_{abs} + C_{sca}. \tag{2.6}$$

Similarly as the cross sections describe, the effect a single particle has on the incident radiation may also describe the effect of a collection of such particles. Consider a collection of particles arranged in such a way that the irradiance on each particle can be assumed equal, then we may simply sum the contributions of each individual particle. This total contribution to absorption is given by the absorption coefficient

$$\kappa = \sum_j N_j C_{abs,j}, \tag{2.7}$$

and equally for scattering by the scattering coefficient

$$\beta = \sum_j N_j C_{sca,j}, \tag{2.8}$$

where N_j are the number of particles of type j per unit volume. More specifically Eq.2.7-2.8 may be rewritten for the specific problem of this thesis, namely the scattering of a collection of spheres. All particles in the collection equal, except a continuous distribution of radii a such that we replace the sum with an integral over radii

$$\kappa = \int_0^\infty N(a)C_{abs}(a)da. \tag{2.9}$$

Substituting the number distribution N with a volume distribution V given by

$$V(a) = \frac{4\pi a^3}{3}N(a) \tag{2.10}$$

gives

$$\kappa = \frac{3}{4\pi}\int_0^\infty \frac{V(a)C_{abs}(a)}{a^3}da. \tag{2.11}$$

By then separating the distribution function V into a probability density function P_{DF} and a scaling factor f given by

$$f = \frac{4\pi}{3}\int_0^\infty a^3 N(a)da = \int_0^\infty V(a)da = \int_0^\infty f P_{DF}(a)da \tag{2.12}$$

which is just the total volume of occupied by all particles per unit volume. Eq. 2.11 then becomes

$$\kappa = \frac{3}{4\pi}f\int_0^\infty \frac{P_{DF}(a)C_{abs}(a)}{a^3}da. \tag{2.13}$$

The scattering coefficient in Eq.2.8 can then be written in the same manner as

$$\beta = \frac{3}{4\pi}f\int_0^\infty \frac{P_{DF}(a)C_{sca}(a)}{a^3}da. \tag{2.14}$$

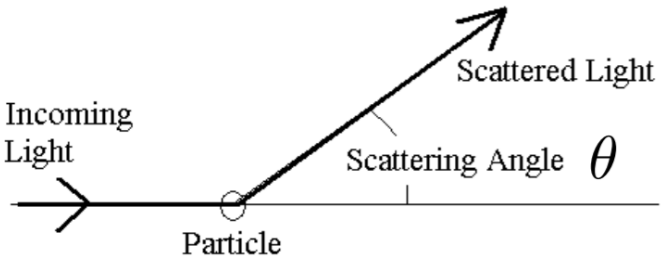

Figure 2.1: scattering by spherical particles (Lelli, 2014)

2.1.2 The phase function

The phase function, $p(\Omega)$, is a dimensionless function describing the angular dependence of the scattered radiation. It may be thought of as a probability density function, which gives the chance of scattering in a given direction. This requires the normalization condition

$$\int_{4\pi} p(\Omega) d\Omega = 1 \tag{2.15}$$

where Ω is the solid angle spanned by the azimuthal angle ϕ and the scattering angle θ. The scattering angle is measured from the direction of incident radiation see Figure 2.1, and the azimuthal angle determines the scattering plane.

The phase function may be defined in terms of the differential scattering cross section which is the energy scattered per unit time into a unit solid angle about a direction Ω' for unit incident irradiance (Bohren and Huffman, 2008). In terms of the scattered irradiance $F_s(\Omega')$ at a distance r from the scatterer and the incident irradiance F_i

$$\frac{dC_{sca}}{d\Omega} = \frac{r^2 F_s}{F_i}. \tag{2.16}$$

The phase function is then

$$p(\Omega) = \frac{1}{C_{sca}} \frac{dC_{sca}}{d\Omega}. \tag{2.17}$$

The asymmetry factor g is the average cosine of the scattering angle

$$g = \int_{4\pi} p(\Omega) \cos(\theta) d\Omega. \tag{2.18}$$

It specifies the degree of asymmetry between forward and backward scattering. The asymmetry factor varies from -1 for pure backscattering ($\theta > \pi$) to 1 for pure forward scattering ($\theta < \pi$).

2.2 Mie theory

Mie, Lorenz-Mie or sometimes even Lorenz-Mie-Debye theory are all different names for the the theory. Mie and Debye worked on scattering by spheres around the same time and independently solved the problem, however they were both not the first to do so. Lorenz was the the first when he published a paper concerning the calculations on the scattering

of radiation by spheres in 1890 (Bohren and Huffman, 2008). However the publication was in danish and did therefore not get a lot of attention at the time. The calculations made by Lorenz where not written in the framework of Maxwells theory of electromagnetism which was included in Mie's treatment of the problem a few year later Mie (1908). In this theis I will use the term Mie theory for brevity.

Mie theory describes the scattering of electromagnetic radiation by spherical particles. In particular plane waves scattering by a spherical particle is exactly described by classical Mie theory in terms of an infinite series (Mie, 1908). Similar solutions are however also known for cylinders, due to the form of the solution this is also sometimes called Mie theory.Worth noting, generalized Mie theory describes spherical particle interaction with arbitrary waves shapes although this is not a focus in this paper.

The Mie theory describes the interaction between a homogeneous sphere and an electromagnetic plane wave. In more detail it describes an incident plane wave interacting with a sphere in a transparent, homogeneous and isotropic surrounding medium. For instance Mie used gold colloids in water as an example. Incident wave induces forced oscillations of charges which are synchronous with the applied field which induces a secondary electric and magnetic field both inside and outside the sphere. The solution then reduces to a boundary value problem for ordinary differential equations. The solution gives the scattered field as a spherical wave composed of two groups of partial waves.

The following eqations are based on the derivation by Bohren and Huffman (2008), for a more complete overview of the derivations see chapter 4 in their book or any of the many other resources on Mie theory.

Denoting the scattered electromagnetic field vector by $(\mathbf{E}_s, \mathbf{H}_s)$ and the incident field $(\mathbf{E}_i, \mathbf{H}_i)$ where the incident plane parallel wave is

$$\begin{aligned} \mathbf{E}_i &= \mathbf{E_0} \mathrm{e}^{i(kx-\omega t)} \\ \mathbf{H}_i &= \mathbf{H_0} \mathrm{e}^{i(kx-\omega t)} \end{aligned} \tag{2.19}$$

The fields must satisfy Maxwell's equation

$$\begin{aligned} \nabla \cdot \mathbf{E} &= 0 \\ \nabla \cdot \mathbf{H} &= 0 \\ \nabla \times \mathbf{E} &= i\omega\mu\mathbf{H} \\ \nabla \times \mathbf{H} &= -i\omega\epsilon\mathbf{E} \end{aligned} \tag{2.20}$$

at all points where where the permittivity ϵ and permeability μ is continuous. Taking the curl of the curl equations above they reduce to Helmholtz equation

$$\begin{aligned} \nabla^2\mathbf{E} + k^2\mathbf{E} &= 0 \\ \nabla^2\mathbf{H} + k^2\mathbf{H} &= 0 \end{aligned} \tag{2.21}$$

where $k = \epsilon\mu\omega^2$. Additionally the internal fields inside the particle (E_{int}, H_{int}) and outside the particle (E_{out}, H_{out}) have to satisfy the following boundary conditions; the tangential components of the electric and magnetic field must be continuous across the boundary of the particle

$$\begin{aligned} (\mathbf{E}_{out}(x) - \mathbf{E}_{int}(x)) \times \hat{\mathbf{n}} &= 0 \\ (\mathbf{H}_{out}(x) - \mathbf{H}_{int}(x)) \times \hat{\mathbf{n}} &= 0 \end{aligned} \tag{2.22}$$

where x is on the surface of the particle and $\hat{n}$ is the outward directed normal to the surface. The electromagnetic fields outside the particle are given by the sum of the incident and scattered field

$$\mathbf{E}_{out} = \mathbf{E}_i + \mathbf{E}_s, \qquad \mathbf{H}_{out} = \mathbf{H}_i + \mathbf{H}_s \tag{2.23}$$

The incident fields $(\mathbf{E}_i, \mathbf{H}_i)$ are known and the internal $(\mathbf{E}_{int}, \mathbf{H}_{int})$ and scattered fields $(\mathbf{E}_s, \mathbf{H}_s)$ are to be determined. Equations and boundary conditions are linear so superposition of solutions is a solution.

In a linear, isotropic, homogeneous medium the electric and magnetic fields must satisfy the wave equation. Two Vector functions, **M** and **N**, are constructed in such a way that they satisfy the requirements for valid electric and magnetic fields. The first of which is chosen to be

$$\mathbf{M} = \nabla \times (\mathbf{c}\psi) \tag{2.24}$$

for scalar function ψ and arbitrary vector **c** Using the fact that the divergence of the curl of any vector function is zero it can be shown that M satisfies the vector wave equation if ψ satisfies the scalar wave equation.

$$\nabla^2\mathbf{M} + k^2\mathbf{M} = \nabla \times \left[\mathbf{c}(\nabla^2\psi + k^2\psi)\right] \tag{2.25}$$

The second vector function **N** is then constructed from **M**

$$\mathbf{N} = \frac{\nabla \times \mathbf{M}}{k}. \tag{2.26}$$

Both **M** and **N** satisfy the vector wave equation, have zero divergence and the curl of each is proportional to the other, so they satisfy all required properties of an electromagnetic field.

The problem of finding the solution to the field equations can therefore be reduced to solving the scalar wave equation. Since our problem involves spherical symmetry it is useful to use spherical coordinates. The generating functions ψ is therefore chosen such that they satisfy the scalar wave equation in spherical coordinates

$$\frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial\psi}{\partial r}\right) + \frac{1}{r^2\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial\psi}{\partial\theta}\right) + \frac{1}{r^2\sin\theta}\frac{\partial^2\psi}{\partial\phi^2} + k^2\psi = 0 \tag{2.27}$$

and the guiding vector **c** from Eq. 2.24 is chosen to be the radius vector **r**. **M** and **N** then become the solutions to the vector wave equation in spherical coordinates. A particular set of solutions to Eq. 2.27 can be found to be

$$\begin{aligned} \psi_{emn} &= \cos(m\phi)P_n^m(\cos\theta)z_n(kr) \\ \psi_{omn} &= \sin(m\phi)P_n^m(\cos\theta)z_n(kr) \end{aligned} \tag{2.28}$$

where $P_n^m(\cos\theta)$ are the associated Legendre functions of the first kind and $z_n(kr)$ is any of the spherical Bessel functions. Due to completeness of these functions any other solution to Eq. 2.27 can be expanded as an infinite series in the functions ψ_{emn} and ψ_{omn} (2.28).

Mie coefficients: a_n and b_n to compute the amplitude of the scattered field. c_n and d_n for the internal field.

$$a_n = \frac{\mu m^2 j_n(mx)[xj_n(x)]' - \mu_1 jn(x)[mxj_n(mx)]'}{\mu m^2 j_n(mx)[xh_n^{(1)}(x)]' - \mu_1 h_n^{(1)}(x)[mxj_n(mx)]'}$$

$$b_n = \frac{\mu_1 j_n(mx)[xjn(x)]' - \mu j_n[mxj_n(mx)]'}{\mu_1 j_n(mx)[xh_n^{(1)}(x)]' - \mu h_n^{(1)}(x)[mxj_n(mx)]'} \tag{2.29}$$

$$c_n = \frac{\mu_1 j_n(x)[xh_n^{(1)}(x)]' - \mu_1 h_n^{(1)}(x)[xj_n(x)]'}{\mu_1 j_n(mx)[xh_n^{(1)}(x)]' - \mu h_n^{(1)}(x)[mxj_n(mx)]'}$$

$$d_n = \frac{\mu_1 m j_n(x)[xh_n^{(1)}(x)]' - \mu_1 m h_n^{(1)}(x)[xj_n(x)]'}{\mu m^2 j_n(mx)[xh_n^{(1)}(x)]' - \mu_1 h_n^{(1)}(x)[mxj_n(mx)]'} \tag{2.30}$$

where prime denotes derivative with respect to the arguments, size parameter x, relative refractive index m, permeability μ_1 and μ of particle and medium respectively. j_n spherical Bessel function (of the first kind) and $h_n^{(1)}$ spherical Hankel function (of the first kind).

2.2.1 Extracting physical quantities

To obtain quantitative results from Mie theory we need to calculate the mie coefficients a_n and b_n and the angular functions π_n and τ_n then sum the series associated with the quantity of interest (Bohren and Huffman, 2008, p.126). Parameters used in radiative transfer depend on a_n and b_n while c_n and d_n are only needed when the internal field is of interest. The internal field is of interest when calculating absorption properties, however this can also be calculated from the scattering and extinction ($C_{abs} = C_{ext} - C_{sca}$) (Mätzler, 2002, p.3-4). Below follows a set of equations which can be used to calculate scattering and absorption properties of the particle from the mie coefficients as derived by Bohren and Huffman (2008). The scattering cross section is given by

$$C_{sca} = \frac{2\pi}{k^2} \sum_{n=1}^{\infty} (2n+1)(|a_n|^2 + |b_n|^2), \tag{2.31}$$

and similarly the extinction cross section is

$$C_{ext} = \frac{2\pi}{k^2} \sum_{n=1}^{\infty} (2n+1)\mathrm{Re}\{a_n + b_n\}. \tag{2.32}$$

The non-zero elements of the scattering matrix (Mueller matrix) for spherical particles, which relates the scattered to incident irradiance of arbitrarily polarized light can be calculated from S_1 and S_2.

$$S_1 = \sum_n \frac{2n+1}{n(n+1)}(a_n \pi_n + b_n \tau_n)$$

$$S_2 = \sum_n \frac{2n+1}{n(n+1)}(a_n \tau_n + b_n \pi_n) \tag{2.33}$$

The relationship between incident and scattered field amplitudes in the far field is then

$$\begin{pmatrix} E_{\parallel s} \\ E_{\perp s} \end{pmatrix} = \frac{e^{ik(r-z)}}{-ikr} \begin{pmatrix} S_2 & 0 \\ 0 & S_1 \end{pmatrix} \begin{pmatrix} E_{\parallel i} \\ E_{\perp i} \end{pmatrix} \tag{2.34}$$

and assumes asymptotic relation.

If we are interested in the polarization of light it is common to use the Stokes vector, which completely describes the polarization state of a light. The Mueller matrix provides the most general and complete description of the response of a medium to excitation by polarized light.

$$\begin{pmatrix} I_s \\ Q_s \\ U_s \\ V_s \end{pmatrix} = \frac{1}{k^2 r^2} \begin{pmatrix} M_{11} & M_{12} & 0 & 0 \\ M_{12} & M_{22} & 0 & 0 \\ 0 & 0 & M_{33} & M_{34} \\ 0 & 0 & -M_{34} & M_{33} \end{pmatrix} \begin{pmatrix} I_i \\ Q_i \\ U_i \\ V_i \end{pmatrix} \tag{2.35}$$

where the matrix elements are given in terms of S_1 and S_2 as follows

$$\begin{aligned} M_{11} &= S_1^2 + S_2^2 \,, & M_{12} &= S_1^2 - S_2^2 \,, \\ M_{33} &= S_2^* S_1 + S_1^* S_2 \,, & M_{34} &= S_2^* S_1 - S_1^* S_2 \,, \end{aligned} \tag{2.36}$$

2.2.2 Mie code

Almost no one used Mie theory before computers, due to computational labor involved in computing each term. This is a likely part of the reason why Mie's paper (Mie, 1908) didn't get a lot of recognition until many years after it was first published. In the 1950s and 60s when computers started to be able to do increasingly complex tasks, Mie's paper started to gain interest. Among the first publicly available Mie codes was the code created by J.V. Dave in 1968 (Wiscombe, 1979). Many possibilities arose as a product of the increased processing capabilities of computers. The processing capabilities of computers, accompanied by an increased interest in light scattering in many fields, contributed to Mie theory becoming popular after years of scarcely being used.

Therefore Mie's paper (Mie, 1908) is sometimes called Dornröschen (sleeping beauty) (Hergert and Wriedt, 2008). However, even with the increased computational capabilities of computers, the development of stable algorithms took some time (Wriedt, 2012). There are some essential points which should be addressed by every Mie algorithm (Cachorro and Salcedo, 1991). A few of these essentail point are how to determine the number N for truncating the Mie series (e.g. Wiscombe criterion), whether the (Riccati-)Bessel functions will be computed by upward or downward recursion, how downward recursion is initialized if it is used and how to structure the algorithm in an efficient way.

Wiscombe described in an interview on scientific programming and his contributions to atmospheric science tool making, that *"Mie series is kind of weird series. It doesn't actually converge. It converges, and then diverges when you keep going because calculations you use to calculate some of the functions there, some of the Bessel functions, start to diverge. One has to cut it off very precisely.* (Flatau, 2013, p.5). Instabilities such as these contributed to early programs having many problems, such as numerical instabilities and overflow errors.

Chapter 3

Neural networks

In this chapter a short introduction to neural networks is given. As artificial neural networks are modelled after biological neural networks, I will start with an introduction to biological neural networks. Then the typical building blocks of an Artificial Neural Network (ANN) will be described along with how they relate to the biological neural networks, which inspired their early development. Lastly a special type of neural network, known as a radial basis function networks, is described.

3.1 Biological neural networks

The human nervous system is built up of billions of neurons (Basheer and Hajmeer, 2000). These neurons are a special type of biological cells, which are responsible for information transfer in the human body. Neurons are composed of three main parts: soma (cell body), axon and dendrites (Jain et al., 1996), as shown in Figure 3.1. Similarly to other cells in the human body, the soma of a neuron contains organelles, such as a nucleus and mitochondria. The difference between neurons and other cells is their signal processing capability. The axon and dendrites are branch like structures which extend for the cell body (soma), these function as signal processing devices. The axon is responsible for transmitting signals from the cell body to other neurons. As the axons are transmitting signals, the dendrites receive these signals and send them on to the cell body. When comparing the biological model to an artificial neural network, we can say hat the axon can be considered the neurons output while the dendrites are the neurons inputs (Gurney, 1997, p.22).

The signal transfers between an axon and dendrite occurs at the synapses, which are at the interface of a neurons axon and the target neurons dendrite. Each synapse, signal transfer, has three components: a presynaptic element, synaptic cleft and a postsynaptic element (Squire et al., 2012, p.64). The presynaptic elements are spezialised parts of the axon, whereas the postsynapic elements are specialized parts the dendrite. The synaptic cleft is a small gap between the two elements, which is of the order of approximatesly 20 nanometers wide (Gurney, 1997, p.23).

Signals that are sent from the presynaptic element are received at the postsynaptic element in the form of neurotransmitters. The neurotransmitters initiate a chemical process. This chemical process changes the electrical potential across the neuron's cell membrane. When the sum of all these membrane potentials exceeds a certain threshold the neuron is said to fire, and an action potential is created which sends new signals to neighboring neurons (Gurney,

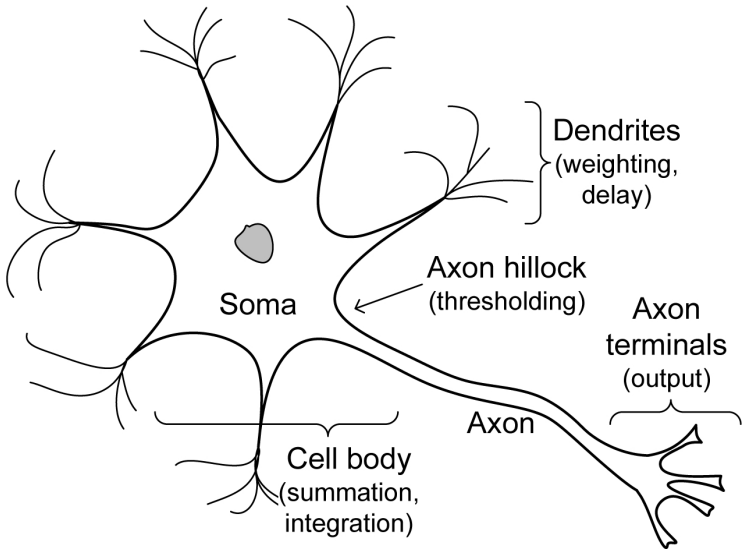

Figure 3.1: Schematic drawing of a biological neuron (Kravtsov et al., 2011)

1997, 25).

The biological neural network structure functions as a massive set of hubs transferring signals all over the network in a seemingly chaotic manner. What it creates is a network with strong collaborations between many units (e.g. cells) enabling successfully processing of complex calculations, situations, choices etc. This process makes the human mind one of the wonders of the biological world. Hence it is has been a established mission to try to recreate the human mind in an artificial setting (e.g. computer processing). The most promising method to do so is Artificial Neural Networks, also known as machine-learning, artificial intelligence etc.

3.2 Artificial neural networks

Artificial neural networks (ANNs) are structures comprised of densely interconnected adaptive simple processing elements, called artificial neurons. These neurons are capable of performing massively parallel computations for data processing and knowledge representation (Basheer and Hajmeer, 2000). The concept of ANNs arose in hopes of harvesting the extraordinary problem solving ability of the human brain. Early ANNs models therefor naturally tried to mimic their biological counterpart (Jain et al., 1996). The first to try this were McCulloch and Pitts (1943) and subsequent ANNs are based on their model.

3.2.1 Artificial neurons

The artificial neuron as it was first introduced by McCulloch and Pitt in 1943 was based on the basic properties of how the biological neuron fires. Thus the artificial neuron follows the threshold-principle based on the biological neurons properties (Maan et al., 2017, p.1734). The artificial neurons as described by McCulloch and Pitt are therefore binary threshold units, which generate an output of one if the weight sum of all inputs exceed a certain threshold. However, if the sum of all inputs do not exceed this threshold, the output wold be zero. Ever since the first artificial neurons were proposed, a number of differing type of artificial neurons

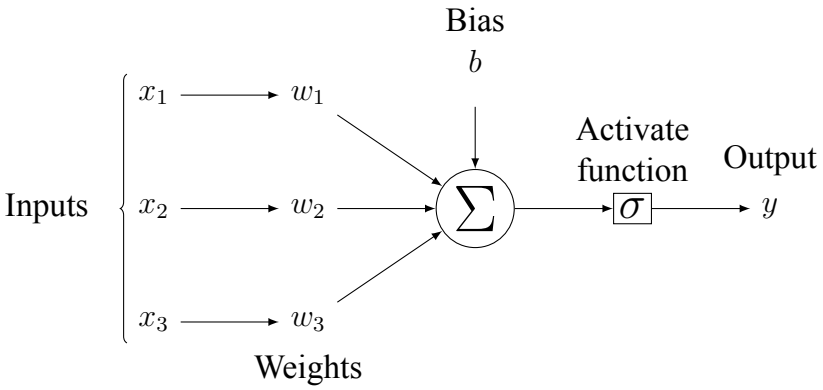

Figure 3.2: Schematic of a generalized McCulloch and Pitts neuron

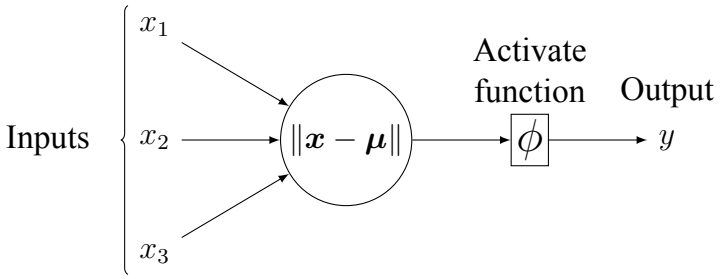

Figure 3.3: Schematic of a RBF neuron

have emerged. The scope of this thesis does not however allow for a detailed account of these. The two examples which I will describe in more detail are the generalized McCulloch and Pitt neuron and Radial basis function neuron.

Generalized McCulloch and Pitt neuron

A commonly used neuron model today is a generalized version of the McCulloch and Pitts neuron. Their inputs are processed in 3 steps; First the inputs x_i are scaled based on a weight parameter w_i. Then the weighted inputs are summed together and a bias is added. Lastly the resulted sum is sent through an activations function σ which produces the output of the neuron. The schematic of a generalized neuon is shown in Figure 3.2.

$$y = \sigma\left(\sum_{i=1}^{n} w_i x_i + b\right) \tag{3.1}$$

The bias can be thought of as the output of a fictitious neuron becoming input x_0 with corresponding weight w_0. Equation 3.1 can then be written in vector form

$$y = \sigma(\boldsymbol{w} \cdot \boldsymbol{x}) \tag{3.2}$$

where $\boldsymbol{w} = [w_0, w_1, \ldots, w_n]$ and $\boldsymbol{x} = [x_0, x_1, \ldots, x_n]$.

Radial basis function neuron

A second type of neuron is the Radial Basis Function neuron, as shown in Figure 3.3. Activation of these neurons depends on the distance between the input vector $\boldsymbol{x}$ and the center of the RBF $\boldsymbol{\mu}$.

$$y = \phi\left(\|\boldsymbol{x} - \boldsymbol{\mu}\|\right) \tag{3.3}$$

As the name suggests the activation function used in these neurons are radial basis functions. By far the most common RBF used in ANNs is the Gaussian function (Marsland, 2015, 113).

$$\phi = e^{-\frac{\|\boldsymbol{x}-\boldsymbol{\mu}\|^2}{\sigma}} \tag{3.4}$$

here the weights represent the center of the bell curve (maximum activation) and σ controls the width.

Each artificial neuron receives inputs either from other neurons or outside stimuli similarly to biological neurons. The result is then sent to connected nodes, which are either other neurons or an output node.

3.2.2 Activation functions

The activation function is one of the most important parts of a neural network. The main purpose of the activation function is to transform the input signal of a neuron to its output signal. In general, activation functions can be categorized into linear and non-linear. Although there are infinitely many possible activation functions, only a few have become popularly used. Non-linear activation functions are the most commonly used activation functions. This is the case for all but the simplest relationships, where the input-output mapping is non-linear. To be able to model non-linear input, a non-linear activation function has to be used. This is due to the fact that a linear combination of linear functions also is linear. In the hidden layers the activation function is used to apply non-linearity to the input-output mapping to make the ANN able to approximate arbitrary functions. This requires a non-linear activation function. However output layer also commonly uses linear activation simply to scale the output. For the hidden layer some of the most common activation functions are the sigmoid, tanh, ReLU and RBF. Output layer also regularly uses linear or softmax activation depending on the type of output the network should provide.

3.2.3 Network architecture

The architecture of a ANN is determined by the interconnection of the neurons. Two commonly used architectures are Feed Forward Neural Network (FFNN) and RNN. Both these network architectures are named after how the information flows through the nodes of the network. In FFNN each neurons input comes from a previous layer. For RNN on the other hand the inputs can be sent back to previous layers to cycle through them again with new processed inputs.

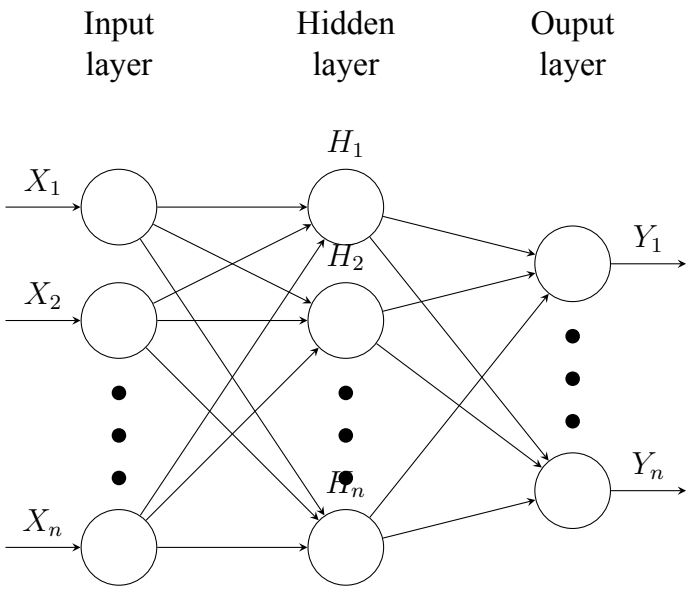

Figure 3.4: Feedforward neural network

The acyclic nature of FFNNs make them the simplest type of network architecture (Marsland, 2015, p.73). Allthoug the structure of a FFNN is simpler than the structure of a RNN, the neural network trained for the purpose of this paper is a FFNN. One of the most common NN is the Mulitlayer Perception (MLP) which is a type of FFNN. A MLP neural network usually consists of a input layer, one or more hidden layers and an output layer (Wang et al., 1999).

Figure 3.4 represents a multilayer perception feed forward neural network. Each node in the hidden layers and output layer of Figure 3.4 represents a artificial neuron. The lines to and from these nodes can be thought of as analogies to the dendrite and axons of the biological neurons respectively. Additionally the weights can represent the synapses and the activation function approximates the activity i the soma (Jain et al., 1996, p.34). In recent years however, a different type of feed forward neural networks have become increasingly popular, the Radial Basis Neural Network (RBFNN).

3.2.4 Radial basis function neural network

The RBFNN is a type of FFNN, and was proposed by Lowe and Broomhead (1988). It is based on the theory for RBFs in multivariable interpolation (Powell, 1987). RBFNNs are three layer FFNN , and structured in the same way as the Mulitlayer Perception Feed Forward Neural Network, Figure 3.4. In contrast to the Mulitlayer Perception (MLP) Feed Forward Neural Network (FFNN) which functions with the basic neuron 3.2, the Radial Basis Neural Network (RBFNN) functions with hidden neurons H given by Figure 3.3. The input layer just distributes the inputs to the various neurons in the hidden layer without any weight factor (unlike in a standard MLP). The hidden layer consists of RBF neurons (Eq. 3.3). The output layer computes the weighted sum of the RBF output (Equation 3.1). The complete input-ouput mapping for a RBFNN is therefore

$$y_j = \sigma\left(\sum_{i=1}^{n} w_{ji}\phi_i\left(\|\boldsymbol{x} - \boldsymbol{\mu}\|\right) + b\right) \tag{3.5}$$

where there are n number of neuron in the hidden layer, k input parameters and j outputs. For RBFNNs the activation function in the output layer $\sigma(\cdot)$ is usually either a linear function if the network is used for function approximation or a sigmoid in the case of a classification problem. For function approximation the equation can then be simplified to

$$y_j = \sum_{i=1}^{n} w_{ji}\phi_i + b, \tag{3.6}$$

and the full output of the RBFNN takes the form

$$y_j = \sum_{i=1}^{n} w_{ji}\exp\left(\sum_{k=1}^{n_i} \frac{(x_k - \mu_{ik})^2}{\sigma}\right) + b_i \tag{3.7}$$

Creating and using a RBFNN involves four main things; determining the number of neurons in the hidden layer, the centers and width of the RBFs and the weights for the output layer. The main difficulty in designing a MLP for a problem is choosing an appropriate number of hidden layers and the size of these layers. In RBFNNs this is seemingly easier as we generally only have one hidden layer which we have to specify the size of. However, due to the localized activation of the RBF neurons, choosing the location and width of the RBFs has a large impact on the performance of the network whereas for MLPs the activation is global.

Due to the capabilities of the RBFNNs over MLP, the former has been chosen in the experiments described in 4.

3.2.5 Neural network training

MLP usually trained with backpropogation algorithms, which is often slow. RBF on the other hand can be trained in many different ways, typically in a two-phase process (Schwenker et al., 2001). The weights of the RBF layer and the output layer are trained separately, which is generally faster than simultaneous training of multiple layers. In the first phase the RBF weights, i.e the centers μ and width σ Eq. 3.3, are trained. The second phase then only trains the weights in the output layer.

3.2.6 Data normalization

Data normalization is often done before the data is used in the training process. While this step is not always necessary it can often help with performance. Normalization involves the rescaling of the data to values appropriate for the model it will be used in. When non-linear output activation functions such as sigmoid or hyperbolic tangent are used, the neural network will only be able to produce outputs in the interval (0,1) and (-1,1) respectively. The desired outputs must therefor be transformed into the appropriate range for the outputs of the network. However even when using a linear output function it may still be useful to transform the outputs as well as the inputs.

For RBF the norm between the input vector and it weight vector (kernel center) determines the activation of the neuron Eq. 3.3. For the most common case of using the Euclidean distance as the norm (Eq. 3.8), different scales can have a large impact on the performance.

$$\|\boldsymbol{x} - \boldsymbol{y}\| = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{3.8}$$

Since every dimension in the input vector is given the same importance in the calculation of the distance, a specific input with much higher variability can easily dominate the result. It is therefor useful to scale the input and output parameters in RBFNNs. There are multiple ways to scale the data, one common way is to log-transform it so that the transformed value x_t is given in terms of the initial value x by

$$x_t = log(x). \tag{3.9}$$

This method is only possible for positive values and is common if they span multiple orders of magnitude.

Chapter 4

Method

The aim of this thesis is to build an artificial neural network to improve the speed of Mie calculations. Previous work has demonstrated that a single layer FFNN can indeed be used to significantly improve the computation time for Mie calculations (Lang (2013)). Based on the findings described in previous chapters regarding neural networks I have chosen to use the RBFNN for this purpose.

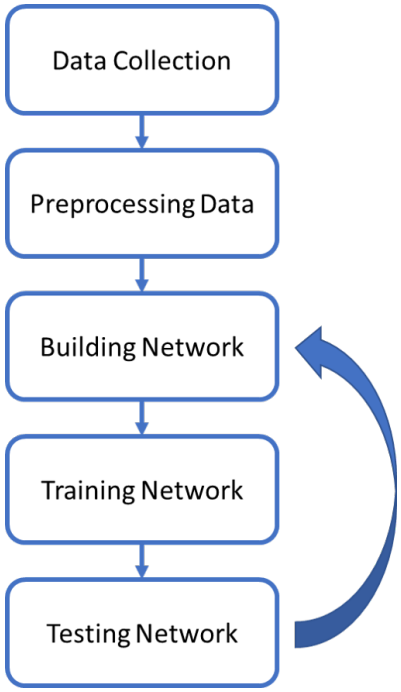

Figure 4.1: Basic flow chart for designing ANN models

The procedure of designing of any neural network can generally be split into 5 basic steps (Al Shamisi et al., 2011), see Figure 4.1. The first step is to collect the relevant data. In this case the data is computer generated by a Mie code. It is necessary to define the input parameters which are to be used in the Mie calculation when generating the data set.

Next the data is preprocessed, which involves possible removal of outliers and scaling/-transforming the data for better training. The third step is building the network with the chosen architecture, RBFNN in this thesis, and selecting the different parameters such as number of neurons. Next the built network is trained on the preprocessed data, where the different weights are adjusted for optimal performance. Lastly the trained network must be

tested again on unseen data to evaluate its generalization capabilities. Depending on the results a new network may be created. In this chapter follows a description of how the Radial Basis Neural Network was created and applied.

4.1 Selecting inputs and outputs

Light scattering by small particles plays an important role in many areas of science. The IOPs of the particles in a medium are necessary for modeling how radiation is transferred in that medium. These include the scattering and absorption coefficient β and κ as well as the scattering phase function p.

$$\frac{\cos(\theta)}{\kappa+\beta}\frac{\mathrm{d}L(\Omega)}{\mathrm{d}z} = -L(\Omega) + \frac{\beta}{\kappa+\beta}\int_{4\pi} p(\Omega')L(\Omega')d\Omega' + S(\Omega) \tag{4.1}$$

Before the radiative transfer equation (Eq. 4.1) can be used the IOPs need to be calculated. This is often done using Mie theory. The particles of interest, such as aerosols in atmosphere, are often large compared to the wavelength of the light used in the calculations. Additionally the calculation time of Mie solution increases with size parameter so a large portion of time is often spend on the calculation of the IOPs.

So the neural network needs to give us some information about these three IOPs. The scattering and absorption coefficients can be used in a straight forward manner in the training of the neural network as they are just single values.The scattering phase function changes at every scattering angle and some more thought has to be put into how this one should be represented. One option is to use values of the phase function at some number of angles and then later interpolate between the points to give a representation for the phase function. Alternatively the reverse option, interpolate first and then train the network on the polynomial coefficients. However, then you may need many points for good representation of phase function. A simple option would be to use the asymmetry factor, g, which may then be use to give an approximation for the phase function from it, such as e.g. Henyey-Greenstein phase function (Henyey and Greenstein, 1941). This only requires the network to give one output related to the phase function, and would presumably make the network easier to train. However, it is important to keep in mind that resulting phase function will be inherently flawed, as it just represents a model of the general shape of these functions. The asymmetry factor was chosen as the output representing the phase function, and other options where tested after the initial consideration.

The inputs for the RBFNN are more straight forward and where chosen from the input variables for the Mie code. The Mie code takes as inputs the wavelength, radius and relative refractive index of the particle and surrounding medium. For scattering by a distribution of different particle sizes the radius will vary so information about the distribution is also necessary. Lastly the calculation of the scattering and absorption coefficients also depend on the volume fraction of particles per volume, this was kept at 1, in order to make the scaling of the coefficient to the desired volume fraction as simple as possible. Since the Mie calculations don't depend on the wavelength directly but only on the relative size of the particle to the wavelength, it was kept constant at 500nm for all calculations. If one is interested in different wavelengths a scaled radius may be used. However the size depends on the particle size distribution.

Aerosol size distributions in the atmosphere are often bimodal and can be approximated by the sum of two lognormal distributions Ahmad et al. (2010). These lognormal distributions depend on the geometric mean radius and the geometric standard deviation. These distributions differ depending on a number of parameters such as, if they are based on a number, mass or volume distribution etc. In this case we use the volume size distribution

$$\frac{dV(r)}{d\ln r} = \frac{V_o}{\sigma\sqrt{2\pi}}\exp\left(-\frac{\ln r - \ln r_{vo}}{\sigma\sqrt{2}}\right), \tag{4.2}$$

where V_o is the volume of particles, r_{vo} the volume geometric mean radius and σ the geometric standard deviation following the notation by Ahmad et al. (2010). The center of the distribution depends on r_{vo}, from here on just referred to as the mean radius, and the width is controlled by σ. These two values were chosen as the input variables describing the size of the particles. This still describes scattering by different wavelengths, but by simply scaling the size to the same amount as the wavelength, the size parameter kan be kept constant.

The relative refractive index which is a complex value is split into it's real and imaginary part. The real part of which is used as inputs.. The refractive index of the surrounding medium is chosen to be unity such that the relative refractive index used in the calculations is simply the refractive index of the particle.

4.2 Generating training data

The data sets containing the input and output parameters discusses in section 4.1 above where generated with the Mie code developed at the University of Bergen, based on the implementation described by Hong Du.The selected upper and lower bounds on the input values where roughly based on the measured values for atmospheric aerosols (Ahmad et al., 2010). The most notable difference is the increase of the maximum mean radius up to 1mm, to include small water droplets. The generated training data sets uses values selected from within the range in Table 4.1 and their corresponding outputs where computed. One attempt was made with a reduced range, where the maximum mean particle radius and imaginary refractive index was lowered to the values in Table 4.2,

Table 4.1: Range of input parameters for generating data set

Input parameters	**Min. Value**	**Max. Value**
Mean Radius	10^{-7}	10^{-3}
Width	0.3	0.5
Real refractive index	1.3	1.7
Imaginary refractive index	10^{-9}	0.5

Table 4.2: Reduced range of input parameters for generating data set

Input parameters	**Min. Value**	**Max. Value**
Mean Radius	10^{-7}	10^{-4}
Imaginary refractive index	10^{-9}	10^{-6}

How the input is selected in the chosen range of values makes a difference, as it changes distribution of inputs for the network to train on. Two different ways were attempted, uniform

and log-uniformly (uniform in the log domain) distributions. It should be mentioned that the natural logarithm was used. For both of these, the same uniformly distributing random number generator was used. Based on these values the data sets was created such that.

4.3 Creating the RBFNN

The RBFNN were generated and trained with Matlab's inbuilt function "newrb" from the neural network toolbox. The best performing RBFNN can be found online (Stremme, 2019). The function requires a data set of inputs and its corresponding outputs, selected based on the purpose of the network described in the section above. New RBF neurons are iteratively added to the network. This is done by calculating the mean square error (MSE) between the RBFNN and the true outputs, and then centering the new RBF neuron on the input vector that minimizes this error. The RBFNN created with

```
net = newrb(P,T,Goal,Spread,MN,DF)
```

where P is the $R \times Q$ input matrix of Q input vectors of length R, similarly T is the $S \times Q$ output matrix containing Q output vectors of length S. Goal represents a stopping criterion for training which is reached when the MSE is lower than the specified goal. The maximum number of neurons MN represents a second stopping criterion. The spread controls the width of each RBF neuron and is related to σ from Equation 3.4 by

$$\sigma = \frac{spread}{\sqrt{-\ln(0.5)}}. \tag{4.3}$$

Lastly DF specifies how many neurons need to be added before the training error display is updated.

The generated RBFNN will minimize the MSE between the networks outputs and the target outputs, corresponding to the true values associated with the given inputs. This error can be made arbitrarily small with a RBF center on each input vector if the widths are small enough not to overlap. The result of this is a "perfect" performance on the training data. However, when this occurs it often result in poor performance on unseen data and is commonly referred to as overfitting. To avoid this certain parameters can be adjusted, such as using a stopping criterion in order to stop training before all parameters are perfect for the training set, or adjusting the spread of the RBF to interpolate between points. To check the effect of these changes, sets of data the trained RBFNN has not seen before must be used for the test set. The error on both the training and test set where taken into consideration when adjusting these values.

4.3.1 Stop criterion

The chosen stop criterion in this case is the maximum number of neurons as opposed to a set MSE. This was based on personal preference to give more control over the size of the networks. Additionally the training data is scaled down and the MSE is calculated on these scaled values, which are very different from the goal of reducing the relative error of the true values.

4.3.2 Spread

Spread is a parameter that needs to be changed to based on how fast the outputs are changing, as well as with the number of neurons. Lower number of neurons generally require larger spreads to fit the whole input space. Finding a good spread value is one of the most difficult parts of creating a RBFNN, and can have a huge effect of its performance. The spread determines the width of the Radial Basis Function, thereby determining the size of the area where the neuron responds with significant outputs. If the spread is too big, a larger number of neurons are required to adapt to a rapidly changing function. Additionally it will generally only be able to represent the general trend of the function, resulting in what is known as underfitting. On the other hand if the spread is too small, more neurons are required for a slowly changing function. This would result in a poor representation of the general trend of the function, even though the error might be low at the training data points, known as overfitting (Wang and Xu, 2013). In these tests, mostly trial and error was used to identify good spreads.

4.4 Number of neurons

In order to create a functional Neural Network training parameters have to be chosen. Similarly to the spread the number of neurons can also have a negative effect on performance if there are either too few or too many. For each selected spread a variety of different neuron numbers where tested. Based on the change in performance with a given set of neurons the next was chosen. If the network stopped performing better or alternatively started to perform worse the spread was lowered and the number of neuron selection started over. As the spread decreased the minimum neuron number was increased based on experience with the previous test, e.g if spread 5, 50 neurons performs poorly and gets better with more neurons than lower spread is also expected to follow this trend.

4.5 Error analysis

After each iteration, parameters were changed in order to see which results the network produced. Several iterations were done and numerous selections of max neuron number and spread were chosen in order to obtain lower error. As results become more promising, that is the error got lower, this narrowed down the parameter window. There are multiple ways of analyzing the performance of the RBFNN, whether to decide if the model works or which changes have to be made. After each new network was trained its performance was checked on the the training data as well as unseen test data, sometimes called validation set. The percentage error (PE)

$$PE_i = 100\frac{(T_i - O_i)}{T_i} \tag{4.4}$$

was calculated for each output in both sets, where the correct target output is T_i and its corresponding network output is O_i. It is worth noting that the values T and O used in the error calculations are the values rescaled. Meaning that the values were scaled back to their true value, before any data transformation was done in the preprocessing step. This is very important as the error on the transformed data may be very different. The mean absolute

percentage error (MAPE) was calculated from the PE

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}|PE| \tag{4.5}$$

as well as the maximum absolute percentage error (MAXAPE)

$$MAXAPE = max(|PE|) \tag{4.6}$$

and the standard deviation (SD)

$$SD = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(PE_i - \overline{PE})^2} \tag{4.7}$$

The error values on these two sets signify possible problems or strength of the network model, e.g. much larger errors on the test set shows that the network does not generalize well to new data.

Chapter 5

Results

5.1 Log-uniform data set

The first data set was generated using a log-normal distributed set of the input parameters from Table 4.1. Figure 5.1 shows the distribution of points as used in the Mie code. The values used in the NN however, where scaled down to similar orders of magnitude by taking the natural logarithm of all parameters. Such that the Figure 5.2 shows the distribution of points that the NN get as inputs. The ith row and column represent the ith input parameters as seen in the order of Table 4.1.

The first set of RBFNN and their performance on the training and test set can be seen in Table 5.1. We can see some initial values for number of neurons and RBF spread used to create the RBFNN and their corresponding mean absolute percentage error (MAPE), maximum absolute percentage error (MAXAPE) and standard deviation of the errors (SD). Further, we can see the increase of number of neurons for a set of different spread values. For each spread the number of neurons were increased until either the improvements in the mean square error (MSE), used as the optimisation parameter flattened out, or until the error on the test sets started getting worse.

For the largest spread values we can see the that the network stops improving already around 150 neurons, while the errors are still high. With too large spread the network can only show a rough trend and not change quickly enough to the actual variation in values, resulting in overfitting. The smaller spreads on the other hand keep reducing the error for over the double amount of neurons, and with an overall lower error. The network seems to generalize well giving similar error for both training and test data set.

The general trend is decreasing MAPE for decreasing spreads with an appropriate number of neurons, this continues until around a spread of 2-3, after which the error increases again. For example comparing the 400 neuron example for spread 2 with the same for spread 1, the MAPE and MAXAPE is significantly larger for the latter. Overall the MAXAPE never seems to go down much lower than around 10-20% for the scattering and absorption coefficient, and around 5-10% for the asymmetry factor. This stays true even on the training data.

Another set of RBFNN with spread 3 but with higher number of neurons is shown in Table 5.2. Here we can see that there is hardly any change in performance and that while the network performs very slightly better on the training set, the performance decreases on the test set. The network tries to fit the training data better at the cost of its generalization ability to the unseen test data.

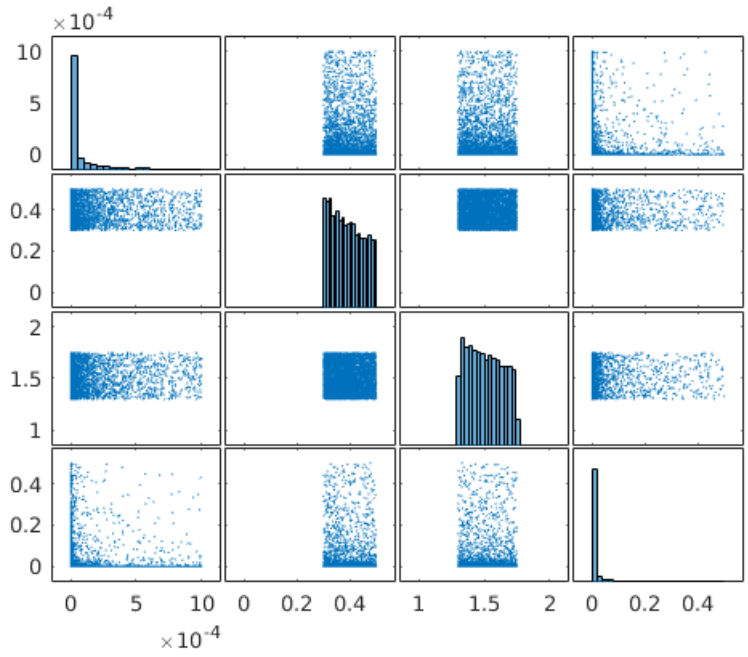

Figure 5.1: Log-uniform data set before log transformation: Distribution of all input plotted against each other. From top to bottom and left to right the inputs are: mean size, width, real and imaginary refractive index.

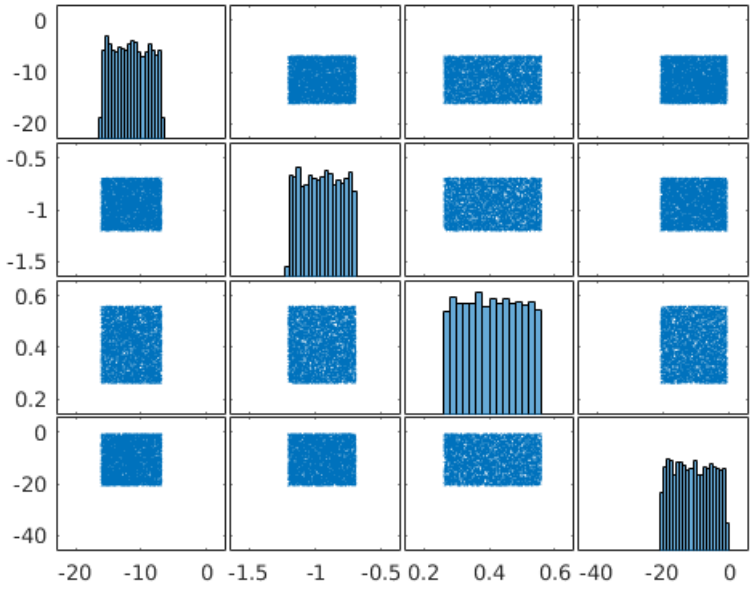

Figure 5.2: Log-uniform data set after log transformation: Distribution of all input plotted against each other. From top to bottom and left to right the inputs are: mean size, width, real and imaginary refractive index.

Table 5.1: Full scale overview for Radial Basis Neural Network training and test data sets for the log-uniform data. Showing max neurons (between 50 and 400) in spread 1, 2, 3, 5, 10 and 20, with results for MAPE, MAXAPE and SD by parameters for scattering, absorption and asymmetry.

			Training set			Test data set		
	Neurons	Parameters	MAPE	MAXAPE	SD	MAPE	MAXAPE	SD
Spread: 20	50	Scattering	10,426	117,693	14,348	10,673	79,119	14,241
		Absorption	6,820	87,562	9,162	7,100	93,293	9,967
		Asymmetry	2,386	38,731	3,726	2,466	28,125	3,797
	100	Scattering	6,750	39,148	8,660	6,774	30,645	8,743
		Absorption	6,028	39,037	7,601	6,161	36,711	7,658
		Asymmetry	2,260	28,893	3,384	2,333	19,540	3,478
	150	Scattering	6,323	33,440	8,094	6,452	42,367	8,296
		Absorption	5,936	34,684	7,474	6,185	33,363	7,679
		Asymmetry	2,179	23,497	3,200	2,276	17,163	3,309
Spread: 10	50	Scattering	9,132	132,804	12,516	9,201	88,440	12,286
		Absorption	5,693	66,347	7,530	5,918	67,650	8,045
		Asymmetry	2,392	33,124	3,620	2,506	25,527	3,705
	100	Scattering	4,215	36,644	5,819	4,305	29,876	5,866
		Absorption	3,412	24,343	4,380	3,335	30,768	4,452
		Asymmetry	1,734	18,105	2,461	1,829	15,957	2,638
	150	Scattering	3,561	24,526	5,049	3,646	23,984	5,087
		Absorption	2,750	17,628	3,604	2,784	16,884	3,617
		Asymmetry	1,289	11,862	1,839	1,362	9,755	1,955
	200	Scattering	3,397	25,699	4,878	3,505	24,642	4,938
		Absorption	2,543	18,072	3,288	2,605	17,040	3,367
		Asymmetry	1,280	11,690	1,831	1,364	9,605	1,965
Spread: 5	50	Scattering	10,538	157,746	16,294	10,298	99,251	14,794
		Absorption	4,728	23,381	6,097	4,705	19,824	5,984
		Asymmetry	2,257	40,268	3,766	2,340	31,049	3,809
	100	Scattering	5,726	82,910	8,361	5,819	66,457	8,397
		Absorption	2,468	19,275	3,243	2,429	16,999	3,180
		Asymmetry	1,772	26,238	2,660	1,824	14,260	2,730
	150	Scattering	3,572	31,694	5,286	3,655	26,916	5,419
		Absorption	2,007	19,045	2,644	1,987	20,028	2,632
		Asymmetry	1,277	15,028	1,886	1,389	11,813	2,042
	200	Scattering	3,108	25,361	4,526	3,131	24,262	4,599
		Absorption	1,561	18,131	2,139	1,556	17,661	2,119
		Asymmetry	1,017	13,096	1,564	1,126	10,401	1,769
Spread: 3	150	Scattering	3,577	44,506	5,426	3,650	27,000	5,503
		Absorption	1,845	18,344	2,481	1,853	15,843	2,441
		Asymmetry	1,229	12,604	1,838	1,357	9,814	2,003
	200	Scattering	2,868	33,131	4,453	2,920	22,936	4,487
		Absorption	1,301	17,596	1,868	1,263	17,814	1,842
		Asymmetry	0,945	10,988	1,458	1,047	8,246	1,616
	250	Scattering	2,463	23,532	3,736	2,521	23,793	3,943
		Absorption	1,142	17,837	1,682	1,131	19,107	1,701
		Asymmetry	0,822	9,353	1,314	0,931	7,637	1,462
	300	Scattering	2,132	18,892	3,290	2,261	23,593	3,582
		Absorption	0,970	17,325	1,518	1,002	20,150	1,595
		Asymmetry	0,752	9,163	1,221	0,870	8,719	1,398
	350	Scattering	1,883	17,025	2,888	2,067	26,054	3,292
		Absorption	0,880	17,276	1,438	0,934	20,564	1,544
		Asymmetry	0,714	9,009	1,139	0,838	9,354	1,336
	400	Scattering	1,637	16,404	2,508	1,871	17,906	2,906
		Absorption	0,810	17,314	1,382	0,861	21,077	1,494
		Asymmetry	0,671	7,807	1,072	0,808	6,368	1,286
Spread: 2	200	Scattering	3,497	55,673	5,094	3,790	26,313	5,472
		Absorption	2,379	29,991	3,356	2,499	16,221	3,443
		Asymmetry	0,869	10,996	1,373	0,955	9,191	1,520
	250	Scattering	2,743	27,630	4,105	3,084	27,362	4,747
		Absorption	1,800	16,911	2,489	1,967	16,700	2,699
		Asymmetry	0,741	10,178	1,207	0,855	7,768	1,413
	300	Scattering	2,025	25,060	3,114	2,244	21,983	3,511
		Absorption	1,344	17,654	1,950	1,399	19,503	2,053
		Asymmetry	0,681	8,197	1,118	0,769	7,491	1,290
	350	Scattering	1,743	24,853	2,980	1,946	18,162	3,097
		Absorption	1,232	17,591	1,829	1,312	19,624	1,929
		Asymmetry	0,637	8,428	1,030	0,750	6,748	1,236
	400	Scattering	1,488	13,850	2,270	1,714	15,603	2,632
		Absorption	1,071	17,005	1,642	1,153	21,933	1,795
		Asymmetry	0,597	8,032	0,950	0,715	6,675	1,160
Spread: 1	200	Scattering	9,758	151,284	14,713	9,486	94,344	14,069
		Absorption	13,467	1532,559	41,492	12,134	86,948	17,261
		Asymmetry	1,704	20,918	2,333	1,919	24,616	2,758
	250	Scattering	7,605	90,217	11,131	7,721	84,887	11,242
		Absorption	8,912	241,585	14,855	9,717	307,791	17,345
		Asymmetry	1,284	24,897	1,831	1,497	17,369	2,214
	300	Scattering	6,019	86,515	8,612	6,375	80,177	9,502
		Absorption	7,501	100,711	11,242	8,179	107,574	12,640
		Asymmetry	1,036	10,054	1,450	1,201	12,736	1,746
	350	Scattering	5,487	57,035	7,787	6,053	66,538	8,786
		Absorption	6,193	77,422	9,008	7,071	113,242	11,000
		Asymmetry	0,830	9,416	1,241	1,004	11,040	1,558
	400	Scattering	4,510	42,283	6,345	5,168	56,098	7,542
		Absorption	5,343	69,613	7,781	6,127	92,695	9,499
		Asymmetry	0,712	7,014	1,094	0,875	14,236	1,437

Table 5.2: Overview of Radial Basis Neural Network training and test data sets for the log-uniform data for spread 3 with higher max neurons (500, 750 and 1000). Results for MAPE, MAXAPE and SD by parameters for scattering, absorption and asymmetry

			Training set			Test data set		
	Neurons	Parameters	MAPE	MAXAPE	SD	MAPE	MAXAPE	SD
Spread 3	500	Scattering	1,316	13,276	2,035	1,499	14,258	2,332
		Absorption	0,739	17,171	1,324	0,816	21,576	1,485
		Asymmetry	0,574	6,556	0,907	0,717	5,816	1,114
	750	Scattering	0,962	11,082	1,495	1,285	20,030	2,093
		Absorption	0,644	16,614	1,234	0,808	21,608	1,509
		Asymmetry	0,450	5,276	0,736	0,629	11,028	1,092
	1000	Scattering	0,858	8,975	1,305	1,219	22,829	1,966
		Absorption	0,605	16,140	1,179	0,821	22,008	1,583
		Asymmetry	0,428	5,031	0,695	0,649	14,428	1,139

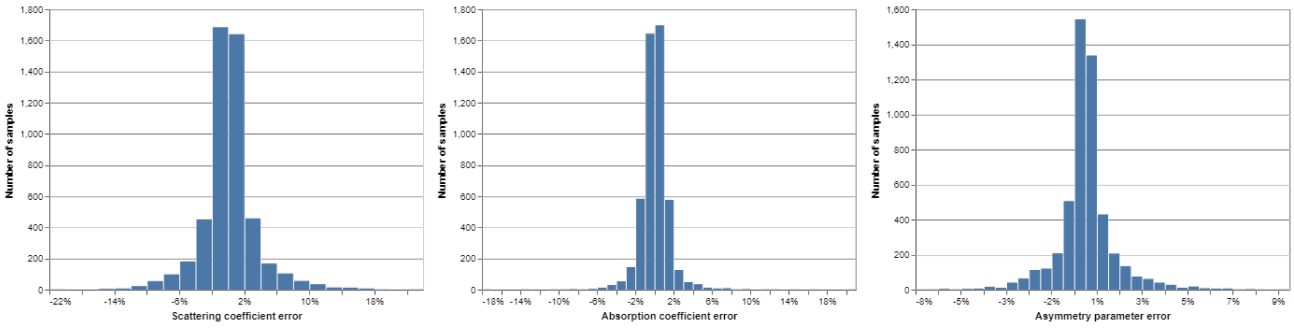

Figure 5.3: Error histogram: From left to right percentage errors for scattering coefficient, absorption coefficient and asymmetry factor. Here all errors, training and test error, combine corresponding to RBFNN with spread 3 and 500 neurons.

Each of the NN created with this log-uniform data set have a Gaussian distribution of errors similar to Figure 5.3. While most of them have some outlier MAXAPE, the standard deviation is still a fairly good representation of the how the errors are distributed. While the figure only shown the combined distribution of errors, the test and training set error look similar individually.

5.1.1 Lower input ranges

A few test where done where the range of mean radius and imaginary refractive index was reduced by lowering their maximum value. The reduced values Table 4.2 where chosen somewhat arbitrarily so that they only vary over 3 orders of magnitude. This was done in order to see if this had any effect on the performance of the network. The results for this data set, as can be seen in Table 5.3, are very similar to the full range Table 5.1. Still having the same problems with high MAXAPE. Once again there is a very clear increase in performance from decreasing the spread, with the best results at spread value of 2. Comparing MAPE for the same number of neurons and spread between this smaller data set and the full one, Table 5.3 and 5.1 respectively, we can see that less neurons are required to achieve a similar error for the smaller set. Additionally we can see that for the best performing networks the MAXAPE for the scattering coefficient is lower for smaller range data set, down from 14-15% to 9-10% on the test set. It is however the opposite for the absorption coefficient, where the MAXAPE

Table 5.3: Radial Basis Neural Network training and test data sets for the log-uniform data set with lower input ranges. Showing max neurons in spread 1, 2, 3, 5, 10 and 20, with results for MAPE, MAXAPE and SD by parameters for scattering, absorption and asymmetry.

			Training set			Test data set		
	Neurons	Parameters	MAPE	MAXAPE	SD	MAPE	MAXAPE	SD
Spread: 20	50	Scattering	4,313	23,896	5,889	4,109	24,135	5,765
		Absorption	2,021	27,706	2,992	1,961	40,155	3,004
		Asymmetry	2,313	17,533	3,126	2,337	15,123	3,138
	100	Scattering	4,300	23,306	5,890	4,115	25,714	5,780
		Absorption	2,020	27,495	2,990	1,958	40,167	3,007
		Asymmetry	2,324	17,669	3,152	2,368	14,824	3,178
Spread: 10	50	Scattering	3,908	29,957	5,560	3,655	23,598	5,320
		Absorption	1,966	26,836	2,910	1,932	39,858	2,967
		Asymmetry	1,802	12,497	2,449	1,806	9,956	2,404
	100	Scattering	3,339	25,729	4,933	3,211	22,932	4,789
		Absorption	1,655	24,810	2,654	1,672	39,013	2,801
		Asymmetry	1,332	11,044	1,865	1,328	8,658	1,845
Spread: 5	50	Scattering	4,204	40,814	5,848	3,735	26,900	5,480
		Absorption	2,068	26,055	3,003	2,013	39,369	3,070
		Asymmetry	1,797	14,946	2,534	1,844	11,248	2,554
	100	Scattering	3,120	24,375	4,415	2,983	16,360	4,210
		Absorption	1,571	25,278	2,598	1,595	38,796	2,778
		Asymmetry	1,119	9,898	1,673	1,118	7,265	1,675
Spread: 3	100	Scattering	2,831	22,618	4,270	2,769	22,424	4,094
		Absorption	1,632	25,287	2,638	1,670	38,721	2,833
		Asymmetry	1,031	9,743	1,543	1,027	7,590	1,532
	150	Scattering	1,915	16,146	2,892	1,904	14,851	2,841
		Absorption	1,507	24,787	2,548	1,569	38,848	2,778
		Asymmetry	0,845	7,680	1,274	0,851	6,811	1,267
	200	Scattering	1,544	13,238	2,306	1,580	13,368	2,348
		Absorption	1,476	24,881	2,521	1,573	38,932	2,795
		Asymmetry	0,769	6,980	1,148	0,770	6,213	1,142
Spread: 2	100	Scattering	2,536	25,775	3,887	2,552	22,373	3,808
		Absorption	1,928	29,006	2,937	2,031	37,880	3,158
		Asymmetry	0,956	8,809	1,452	0,936	8,412	1,424
	150	Scattering	1,694	19,679	2,578	1,668	12,694	2,495
		Absorption	1,590	26,926	2,608	1,673	38,195	2,850
		Asymmetry	0,761	7,386	1,167	0,767	8,126	1,162
	200	Scattering	1,287	12,169	1,961	1,289	9,887	1,948
		Absorption	1,489	25,393	2,529	1,554	38,586	2,772
		Asymmetry	0,672	6,233	1,016	0,682	7,517	1,028
	250	Scattering	1,110	10,043	1,714	1,142	9,422	1,797
		Absorption	1,452	25,324	2,498	1,550	39,026	2,800
		Asymmetry	0,616	6,070	0,940	0,639	6,972	0,972
	300	Scattering	1,002	8,551	1,528	1,060	9,993	1,652
		Absorption	1,423	24,678	2,478	1,542	39,057	2,809
		Asymmetry	0,571	5,515	0,872	0,606	6,555	0,934
Spread: 1	100	Scattering	5,257	54,149	7,457	5,183	50,704	7,393
		Absorption	4,083	30,077	5,653	4,264	33,188	6,041
		Asymmetry	1,140	11,766	1,648	1,151	9,299	1,612
	150	Scattering	3,452	33,729	4,919	3,474	22,830	4,797
		Absorption	2,988	25,827	4,118	3,188	38,123	4,481
		Asymmetry	0,808	7,442	1,219	0,813	6,800	1,212
	200	Scattering	2,042	20,356	2,783	2,082	20,608	2,900
		Absorption	2,407	27,454	3,505	2,626	38,179	3,954
		Asymmetry	0,632	6,268	0,990	0,634	7,461	0,995
	250	Scattering	1,551	15,404	2,146	1,667	16,591	2,315
		Absorption	2,038	27,234	3,052	2,215	38,545	3,414
		Asymmetry	0,531	5,875	0,832	0,558	6,389	0,888

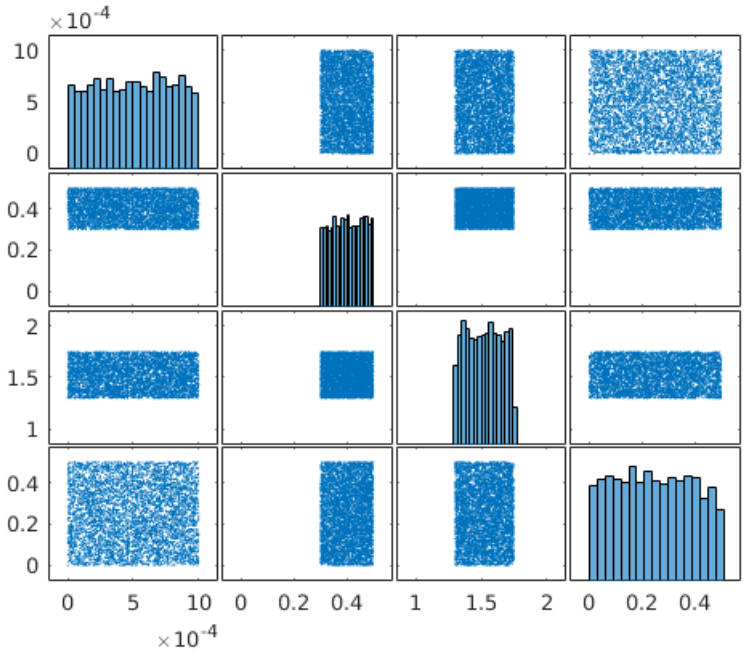

Figure 5.4: Uniform data set before log transformation: Distribution of all input plotted against each other. From top to bottom and left to right the inputs are: mean size, width, real and imaginary refractive index.

increases from around 20% to almost 40%.

5.2 Uniform data set

Similarly to the log-normally distributed data set the uniformly distributed data set was also generated on the full range of inputs as can be seen in Table 4.1. The distribution of points before and after log-transformation an be seen in Figure 5.4 and 5.5. We can see that the mean radius and imaginary refractive index, which ranger over multiple orders of magnitude, are very affected by the choice of distribution. While the inputs are uniformly distributed initially (Figure 5.4), after the log transform (Figure 5.5). We can see that there are actually very few inputs with low mean radius and imaginary refractive index. Especially on the plots where distribution of size and imaginary refractive index is shown simultaneously, we see that this set is prone to outliers. The results in Table 5.4 illustrates the problem this creates nicely, with a very large discrepancy between the performance on the training and test set. The errors on the training set are much lower than for the log-uniform training sets. On the training set the RBFNN with 150 neurons and spread of 3 has a mean error under 0.04%, max of 0.7% for the scattering and absorption coefficients and the asymmetry factor 10 times more accurate than those. Even the smallest networks that where tested with only 50 neurons perform well on the training data. However, the max error on the test set is much higher with at least one outlier of over 100% for each RBFNN.

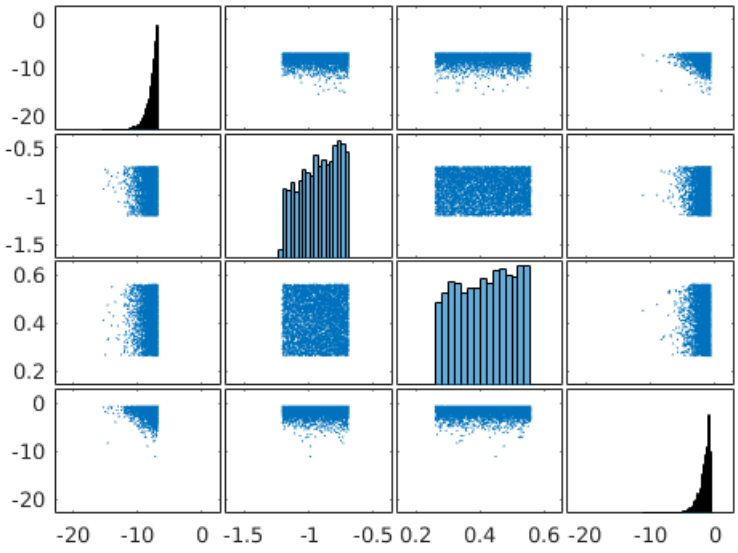

Figure 5.5: Uniform data set after log transformation: Distribution of all input plotted against each other. From top to bottom and left to right the inputs are: mean size, width, real and imaginary refractive index.

Table 5.4: Full scale overview for Radial Basis Neural Network training and test data sets for the uniform data. Showing max neurons (between 50 and 150) in spread 3, 5, 10 and 20, with results for MAPE, MAXAPE and SD by parameters for scattering, absorption and asymmetry.

			Training set			Test data set		
	Neurons	Parameters	MAPE	MAXAPE	SD	MAPE	MAXAPE	SD
Spread: 20	50	Scattering	0,12376	6,26960	0,24946	0,21832	77,91900	2,50710
		Absorption	0,16227	6,00180	0,30560	1,07050	874,09100	27,68440
		Asymmetry	0,06334	1,83540	0,09849	0,12396	58,08480	1,84500
	100	Scattering	0,11679	5,64930	0,22567	5,77630	5564,43000	176,06600
		Absorption	0,13468	3,26050	0,22386	0,27173	99,52940	3,35370
		Asymmetry	0,05716	1,58960	0,08574	0,15489	92,68040	2,93820
Spread: 10	50	Scattering	0,10872	4,25900	0,19571	4,60460	4370,35320	138,31360
		Absorption	0,12989	3,94620	0,22589	0,23954	54,56290	2,29960
		Asymmetry	0,05005	0,85297	0,07146	0,12676	73,18790	2,31750
	100	Scattering	0,05943	2,32330	0,10559	260,48780	260064,88890	8228,08620
		Absorption	0,05853	1,09720	0,10060	0,19056	99,99950	3,29470
		Asymmetry	0,01911	0,42543	0,02787	0,11775	91,47540	2,89850
Spread: 5	50	Scattering	0,11385	4,40840	0,21641	0,27931	95,64700	3,61450
		Absorption	0,11610	4,08850	0,20929	0,30568	153,32500	4,97300
		Asymmetry	0,05063	0,97277	0,07597	0,08949	32,80160	1,05000
	100	Scattering	0,04327	0,91617	0,07574	1,05400	912,59290	29,00660
		Absorption	0,04871	0,94529	0,08584	0,17941	99,77090	3,27810
		Asymmetry	0,00769	0,15255	0,01129	0,09057	77,04140	2,44190
	150	Scattering	0,03963	0,65958	0,06692	0,63189	509,89650	16,28740
		Absorption	0,03959	0,70832	0,06801	0,17513	99,83850	3,29940
		Asymmetry	0,00528	0,08349	0,00744	0,09740	83,86210	2,66050
Spread: 3	100	Scattering	0,04832	0,65184	0,07923	0,22340	96,65240	3,74120
		Absorption	0,05485	0,69602	0,08689	0,22632	146,83350	4,69960
		Asymmetry	0,00626	0,11281	0,00092	0,03975	26,19000	0,84499
	150	Scattering	0,03700	0,67411	0,06305	0,20786	83,60680	3,65460
		Absorption	0,03769	0,63383	0,06481	0,43585	367,21920	11,64420
		Asymmetry	0,00316	0,02335	0,00393	0,05316	41,97560	1,34210

5.3 Combined data set

The last data set used for training the RBFNN was a combination of the uniform set, Figure 5.5, and log-uniform set Figure 5.2.The results of the combined data set can be seen in Table 5.5.

Table 5.5: Overview for Radial Basis Neural Network training and test data sets for the combined set of log-uniform and uniform data. Showing max neurons (between 100 and 300) in spread 2, 3, 5 and 10, with results for MAPE, MAXAPE and SD by parameters for scattering, absorption and asymmetry.

			Training set			Test data set		
	Neurons	Parameters	MAPE	MAXAPE	SD	MAPE	MAXAPE	SD
Spr.: 10	100	Scattering	2,806	26,958	4,530	2,862	30,737	4,561
		Absorption	2,904	18,652	4,023	2,932	18,325	4,058
		Asymmetry	1,096	16,183	1,949	1,148	14,037	2,066
Spr.: 5	100	Scattering	3,216	56,172	5,630	3,289	48,236	5,611
		Absorption	1,922	18,850	2,804	1,924	15,178	2,791
		Asymmetry	1,040	20,838	1,934	1,064	16,667	2,018
Spread: 3	150	Scattering	2,417	61,945	4,399	2,440	34,639	4,432
		Absorption	1,336	23,793	2,016	1,307	17,615	1,979
		Asymmetry	0,731	13,825	1,388	0,784	11,321	1,485
	300	Scattering	1,242	25,646	2,557	1,354	25,974	2,838
		Absorption	0,591	17,587	1,107	0,606	19,796	1,149
		Asymmetry	0,421	8,985	0,886	0,474	7,701	1,011
Spr.: 2	300	Scattering	1,421	20,642	2,410	1,497	21,456	2,638
		Absorption	1,155	16,027	1,729	1,198	17,612	1,828
		Asymmetry	0,412	8,183	0,825	0,463	7,591	0,952

5.4 Closer look at the errors

Most calculations by the RBFNNs are very accurate however the presence of those few high errors motivated a closer look at trying to find where those are located. The absolute value of the percentage error for the 500 neuron RBFNN from Table 5.2 was color mapped onto a scatter plot of the log-transformed values of imaginary refractive index and mean radius of the particle size distribution. The results for the error in scattering coefficient, absorption coefficient and asymmetry factor can be seen in Figure 5.6, 5.7 and 5.8 respectively. For the scattering and asymmetry error there is a clear trend of the largest errors being between a mean radius of e^{-16} and e^{-14} which is exactly on the order of the wavelength used to compute the outputs ($500 \cdot 10^{-9}$). The large absorption errors are clustered similarly, however a bit maybe a bit more towards slightly larger mean radius and also towards mostly small imaginary refractive index values.

Similar plots for the for the uniform data set is not as interesting as it only shows the high errors on the outliers far away from the main cluster of points in Figure 5.5. The error on the training set here is not large due to almost all the points being clustered around large values of the input parameters which the RBFNN is good at approximating. The few small values, of size and imaginary refractive index, that are in the set are calculated exactly by centering a RBF directly on the point during training. Since the points are to far apart however those type of points that are in the test set may end up not being activated at all by any of the neurons and result in a huge MAXAPE.

5.5 Computation time

Computation time was compared between the c++ Mie code and RBFNN from Table 5.2. The c++ Mie code was run with a varying mean radius for the particle size distribution, with a width of 0.4, real part of the refractive index 1.5 and imaginary part 10^{-4}. Each different computation was timed, which resulted in the a variety of different times for similar particle sizes. This was likely due to the program dynamically increasing the number of quadrature points for the integration over particles sizes if the desired accuracy was not reached. The plotted times in Figure 5.9 are the average time for collection of different mean radii. The c++ Mie code increases approximately linearly on the log-log scale, ranging from approximately $5 \cdot 10^{-4}$s for the smallest mean radius to about 100s for the largest. The calculation time of the RBFNN on the other hand does not vary with the input parameters, staying constant around at around $8 \cdot 10^{-3}$s. As a result the RBFNN is slightly slower for very small particles, with a dramatic improvement in speed as the radius is increased, by about a factor of 10000. The point at which the RBFNN becomes equally fast as the c++ code is when the mean radius is around $10^{-6.4}$ which is approximately $4 \cdot 10^{-7}$.

The right plot of Figure 5.9 demonstrates the change in computation time between RBFNN with 500, 750 and 1000 neurons. The RBFNNs with more neurons are slightly slower, with an increase of less than $2 \cdot 10^{-3}$s from 500 to 1000 neurons.

The first 1-2 calculations by the RBFNN where constantly slower than the rest. They were fed the input data in order from smallest to larges mean radius in Figure 5.9. So this higher value for the smallest radii has nothing to do with the actual input parameters, but rather an artifact of how the time measurement was done.

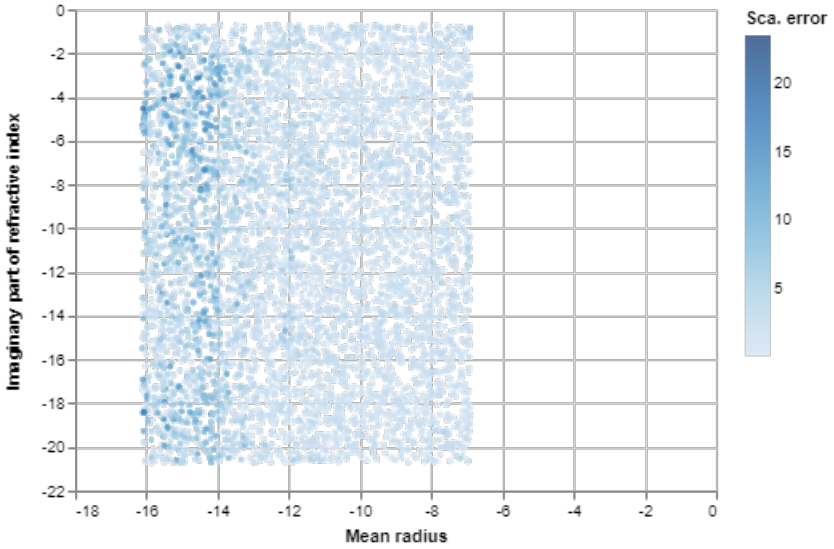

Figure 5.6: Distribution of scattering coefficient absolute percentage error.

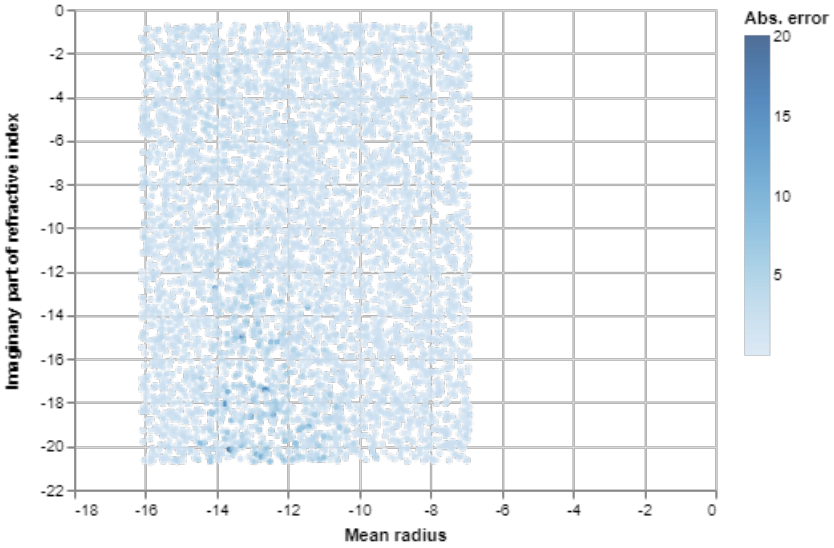

Figure 5.7: Distribution of absorption coefficient absolute percentage error

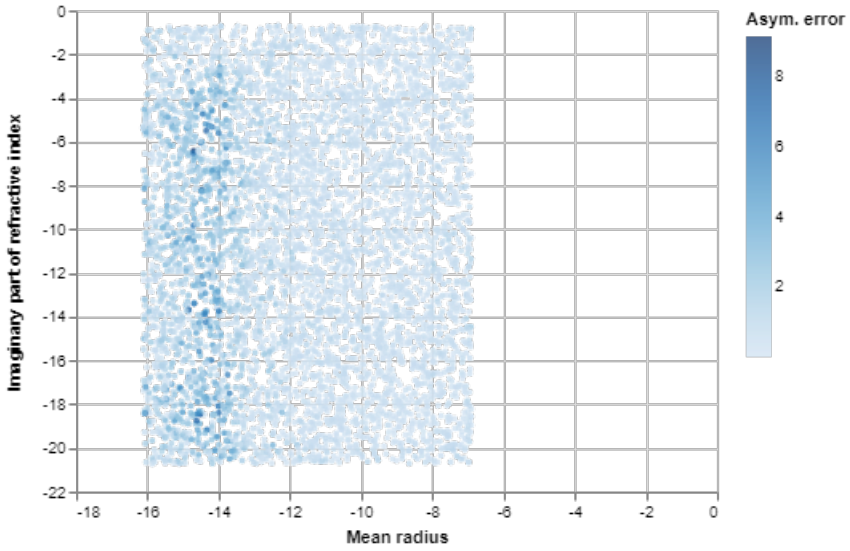

Figure 5.8: Distribution of asymmetry factor absolute percentage error

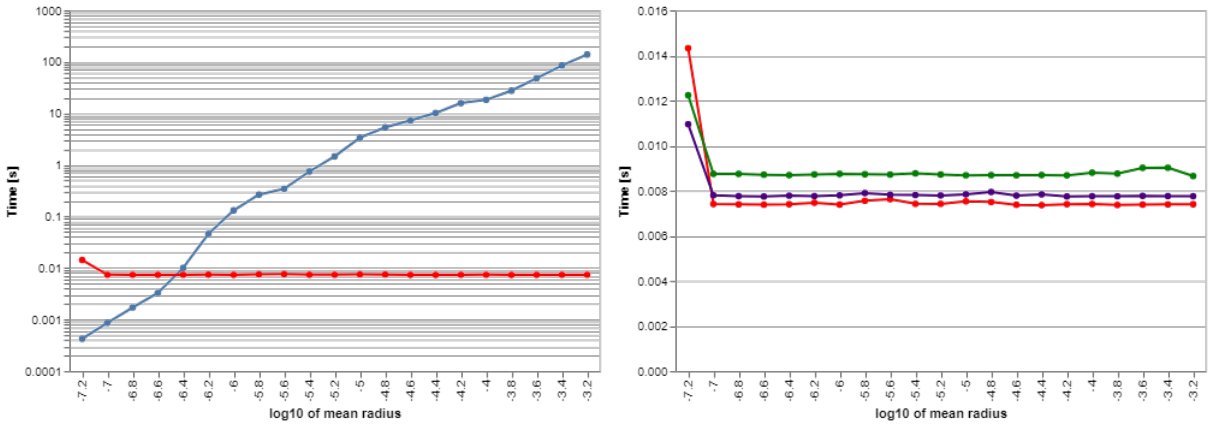

Figure 5.9: Average computation time for particle size distributions of varying mean radius. The mean radius is given in meters and its log base 10 value is shown in the Figure. The initial higher value for the red, purple and green line, before it flattens out, is due to being the first values given to the neural network. Left: Blue line represents the c++ Mie code, red line neural network with 500 neurons. Right: Red, purple and green represent neural network with 500, 750 and 1000 neurons respectively

Chapter 6

Discussion

One of the first questions that came up when I started working on the neural networks was how best to sample the training data. As the ranges for the various inputs (Table 4.1) differ by multiple orders of magnitude, the decision was made early on to log-transform the values in order to scale them to comparable ranges. The first data set (see Figure 5.2) followed a log-uniform distribution, such that the input vectors are evenly spread, which Lang (2013) demonstrated worked well for a regular FFNN with tanh activation function. This turned out to also work fairly well for RBFNN. The RBFNN in Table 5.1 and 5.2, specifically those with spread of 2-3 and more than 300 neurons, achieve similar or for some even slightly better MAPE error than was previously reported by Lang (2013). While a mean error of about 1% is likely good enough in many applications, the presence of a few high maximum errors had to be investigated more.

Initially no exploratory visualisation was done and only a few of the largest errors where manually investigated. During the training attempts on the log-uniform data set these individual values I looked at seemed to often be input vectors with large imaginary refractive index. This led me to wonder if the low number of data points for high values of mean size and imaginary part before the log transformation (see row 1 and 4 Figure 5.1) caused this. In respond which the uniform data set was created. It could have easily been seen that this is not the solution had I earlier spent more time on visualizing the different data set alongside the computed errors like in Section 5.4.

The uniform data set while evenly distributed before the transformation is almost entirely distributed around the top range of the inputs, with a bunch of outlier points, after the transformation. As the transformed data was what the RBFNN was trained on, in hindsight this should have been expected to not perform well. Since outliers are generally bad for RBFNNs as the RBFs need to be spread across the entire input space to be able to generalise, however with outliers typical training algorithms don't work well. Even though the RBFNN trained on this data set weren't able to generalise well over the whole input range, due to a lack of points for smaller input values, the test results illustrated two important points. Firstly the importance of a good distribution of data points and that the large values where not cause of the high MAXAPE.

The main problem of higher error was found to be due to small values of the mean radius of the particle size distribution. This approximately where the mean radius corresponds to a size parameter of 1, and the output vary the most. Distribution of input parameters such that the points are more densely spaced where the output varies rapidly would make sence. So maybe a data set similar to the log-uniform set with even more points around these mean

radii, could help reduce the maximum error. Or potentially creating multiple networks, one for the only smallest mean sizes where the range of is fairly small maybe one orders of magnitude, and then another for the rest where it seems to be easier to fit the NN.

The scaling of the input parameters is usually advantageous since the output of the RBF is based on a distance measure between the input vector and the RBF center (Eq. 3.3). If the size of the different input parameters are not comparable, then the feature with the largest ranges could completely dominate over those with smaller ranges. However there are many ways to scale the data, maybe there is some other method used instead of or in combination with log-transform to produce a data set that is particularly easy to train a RBFNN on.

Different ways to determine RBF centers and training algorithm. Matlabs newrb function centers the RBFs directly on one of the various inputs. Training algorithm adds new RBFs such that they reduce MSE which may not be the best way to do it. Additionally same spread value was used for each RBF neuron in a given RBFNN, the training algorithm used in this paper did not change the spreads at all. As the spread is one of the most important parameters in a RBFNN, allowing for different spread throughout the input space likely could increase approximation performance. The possibility of retrain the RBFNN was considered in order to change the center and width of each RBF, but in the end no attempted where made as this added another whole new topic of various training algorithms to consider. However i suspect such a retraining method is likely to be more effective in combination with a different technique for RBF center initialization is used. As the newrb method for center selection may place them in a local minima that new training algorithms don't manage to move away from.

Asymmetry factor was constantly the most accurate of the output parameters to approximate. It is however also the parameter with the least variation of possible values. The asymmetry factor is by definition limited to values between -1 and 1, and in our case varies even less. This is due to the particle sizes are all comparable with or much larger than the wavelength, where forward scattering dominates. Maybe more information is possible to be extracted. An interesting possibility is to replace the asymmetry factor with a values for the phase function at a set of different angles, to possibly get a more accurate angular distribution of the scattering. Alternatively even do both and use the asymmetry factor for the general curve and use the other outputs as a way modify phase function approximated by the asymmetry factor.

Computation time of traditional Mie depend on multiple parameters. The mean radius was varied in Figure 5.9 which show the expected increase with size, larger size parameters require more terms in the Mie series. However other effects such as the width also play a role, large width of the size distribution requires more large particles to be calculated which take the most time. Additionally the imaginary part of the refractive index can also play a role in computation time. The results in Figure 5.9 show that average computation time of the given input parameters, that is constant width and refractive index, is up to 10000 times slower than the created RBFNN. It should however be noted that depending on the particles of interest this difference will can be significantly less or even more

For each data set many networks are tested against the same test data, such that picking the best performing network from the test performance may not actually be the best one to generalize. This is something that should be considered in the future when trying to fine tune the best parameters. A validation set should be used to test the individual models, and then the selected model should be tested on a new test set which is completely independent. This can be important as there is the possibility of having a good performance on the validation

set purely by chance, without it actually being the NN that generalises best overall. So its worth noting that even though the spread 3 and 500 neuron network performs best on the test set, it has not been thoroughly tested on other data sets that are guaranteed to be completely independent, this remains to be done in the future. The network as it was trained here can be found online (Stremme, 2019).

Chapter 7

Conclusion and outlook

The a single radial basis function neural network with 300-500 neurons is enough to approximate the scattering coefficient, absorption coefficient and asymmetry parameter with an mean absolute percentage error of less than 2%. Depending on the range of and distribution of input the network is trained on smaller network could also give similar results. The computational difference a 500 and 1000 neuron RBFNN is however very small, with the larger network only being $2 \cdot 10^{-3}$s slower per calculation. The time save of a neural network can be enormous for particles much larger than the wavelength, with a potential improvement of 10000 times faster than traditional Mie codes.

The main problem encountered was the potential of outlier errors of up to 20-30% exists for the best performing RBFNN. Current results show that IOP for particle distributions for large size parameter are easier for the RBFNN to approximate than for the ones size parameters close to one. Potentially creating multiple RBFNN which are each trained for different ranges of input parameters may be a way to reduce the error even further. Additionally different training algorithms for the selection of RBF center are recommended to investigate.

Between the output parameters used in the RBFNN (scattering coefficient, absorption coefficient and symmetry factor) the asymmetry factor is the parameter most accurately calculated. It would be interesting to see if RBFNN can be trained to output a more accurate description of the phase function. This would likely require a much larger number of output parameters. Investigating how the RBFNN performance changes with increasing number of output parameters and seeing if such a network could potentially predict atmospheric phenomena such as the rainbow.

Bibliography

Ahmad, Z., B. A. Franz, C. R. McClain, E. J. Kwiatkowska, J. Werdell, E. P. Shettle, and B. N. Holben (2010). New aerosol models for the retrieval of aerosol optical thickness and normalized water-leaving radiances from the seawifs and modis sensors over coastal regions and open oceans. *Applied Optics 49*(29), 5545–5560. 21

Al Shamisi, M. H., A. H. Assi, and H. A. Hejase (2011). Using matlab to develop artificial neural network models for predicting global solar radiation in al ain city–uae. In *Engineering education and research using MATLAB*. InTech. 19

Basheer, I. A. and M. Hajmeer (2000). Artificial neural networks: fundamentals, computing, design, and application. *Journal of microbiological methods 43*(1), 3–31. 11, 12

Bohren, C. F. and E. E. Clothiaux (2006). *Fundamentals of atmospheric radiation: an introduction with 400 problems*. John Wiley & Sons. 3, 4

Bohren, C. F. and D. R. Huffman (2008). *Absorption and scattering of light by small particles*. John Wiley & Sons. 6, 7, 9

Cachorro, V. and L. Salcedo (1991). New improvements for mie scattering calculations. *Journal of electromagnetic waves and applications 5*(9), 913–926. 10

Flatau, P. J. (2013). Interview with warren wiscombe on scientific programing and his contributions to atmospheric science tool making. *arXiv preprint arXiv:1304.1582*. 10

Gurney, K. (1997). *An introduction to neural networks*. CRC press. 11

Henyey, L. G. and J. L. Greenstein (1941). Diffuse radiation in the galaxy. *The Astrophysical Journal 93*, 70–83. 20

Hergert, W. and T. Wriedt (2008). Mie theory 1908-2008. *Present Development and Interdisciplinary Aspect of Light Scattering*, 15–17. 10

Jain, A. K., J. Mao, and K. M. Mohiuddin (1996). Artificial neural networks: A tutorial. *Computer 29*(3), 31–44. 11, 12, 15

Jiang, K., X. Han, and K. F. Ren (2012). Scattering from an elliptical cylinder by using the vectorial complex ray model. *Applied optics 51*(34), 8159–8168. 4

Kravtsov, K., M. P. Fok, D. Rosenbluth, and P. R. Prucnal (2011). Ultrafast all-optical implementation of a leaky integrate-and-fire neuron. *Optics express 19*(3), 2133–2147. ix, 12

Lang, A. V. (2013). Software development for radiative transfer applications. 2, 19, 37

Lelli, L. (2014). Aerosol and Clouds kernel description. ix, 3, 6

Lowe, D. and D. Broomhead (1988). Multivariable functional interpolation and adaptive networks. *Complex systems 2*(3), 321–355. 15

Maan, A. K., D. A. Jayadevi, and A. P. James (2017). A survey of memristive threshold logic circuits. *IEEE transactions on neural networks and learning systems 28*(8), 1734–1746. 12

Marsland, S. (2015). *Machine learning: an algorithmic perspective*. CRC press. 14, 15

Mätzler, C. (2002). Matlab functions for mie scattering and absorption, version 2. *IAP Res. Rep 8*(1). 9

McCulloch, W. S. and W. Pitts (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics 5*(4), 115–133. 12

Mie, G. (1908). Beiträge zur optik trüber medien, speziell kolloidaler metallösungen. *Annalen der physik 330*(3), 377–445. 1, 7, 10

Nussenzveig, H. M. s. (1977). The theory of the rainbow. *Scientific American 236*(4), 116–128. 1

Powell, M. J. (1987). Radial basis functions for multivariable interpolation: a review. *Algorithms for approximation*. 15

Schwenker, F., H. A. Kestler, and G. Palm (2001). Three learning phases for radial-basis-function networks. *Neural networks 14*(4-5), 439–458. 16

Squire, L., D. Berg, F. E. Bloom, S. Du Lac, A. Ghosh, and N. C. Spitzer (2012). *Fundamental neuroscience*. Academic Press. 11

Stremme, M. J. (2019). Code for radial basis function neural network. 22, 39

Wang, F., V. K. Devabhaktuni, C. Xi, and Q.-J. Zhang (1999). Neural network structures and training algorithms for rf and microwave applications. *International Journal of RF and Microwave Computer-Aided Engineering 9*(3), 216–240. 15

Wang, H. and X. Xu (2013). Determination of spread constant in rbf neural network by genetic algorithm. *Int. J. Adv. Comput. Technol.(IJACT) 5*(9), 719–726. 23

Wiscombe, W. J. (1979). *Mie scattering calculations: Advances in technique and fast, vector-speed computer codes*. National Technical Information Service, US Department of Commerce. 10

Wriedt, T. (2012). Mie theory: a review. In *The Mie Theory*, pp. 53–71. Springer. 1, 10