TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

DIPLOMARBEIT

Simulation von magnetischer Ultra-kleinwinkelneutronenstreuung mittels Streudatensynthese

Ausgeführt am Atominstitut

der Technischen Universität Wien

unter der Anleitung von

Univ. Prof. Dipl. Ing. Dr. Gerald Badurek

und Dipl. Ing. Dr. Erwin Jericha

durch

Alexander Zdarzil

Erzherzog-Karl-Straße 78/20

1220 Wien

März 2014

Alexander Zdarzil

Erklärung zur Verfassung der Diplomarbeit

Hiermit erkläre ich, dass ich diese Arbeit selbstständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen – die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Ort, Datum Unterschrift

Kurzfassung

Die vorliegende Arbeit aus dem Bereich Neutronenphysik befasst sich mit einer Aufgabenstellung zum Thema magnetische Ultrakleinwinkelneutronenstreuung (USANSPOL). Es wird gezeigt, wie durch die Erstellung numerisch berechneter (synthetischer) Streudaten und ihrer anschließenden Auswertung Rückschlüsse für die Analyse von realen Streudaten gewonnen werden können. Zentraler Gegenstand ist die Entwicklung einer auf der Softwareumgebung IGOR Pro basierenden „USANS Simulation Application“, mit der unterschiedlich parametrisierte Streukurven rasch über eine einfach zu bedienende Oberfläche erstellt werden können. Die Arbeit beginnt nach einem einführenden Überblick über das Themengebiet mit einer mathematischen Abhandlung der theoretischen Grundlagen, die später das Fundament der Applikationskernbereiche bilden. Im anschließenden Hauptteil wird die Vorgehensweise, wie die Streudatensynthese umgesetzt wurde, dokumentiert: Eingangs wird die verwendete Software und ihre Schlüsselkonzepte präsentiert, danach werden alle Entwicklungsschritte bis hin zur Erstellung einer kompletten USANSPOL Streukurve detailliert beschrieben. Für jeden Teilbereich wird das mathematische Konzept vorgestellt, die programmtechnische Umsetzung beschrieben, die Bedienung der zugehörigen Oberflächenelemente erläutert und anschließend ein exemplarisches Ergebnis abgebildet. Lösungsansätze für eine Vielzahl an auftretenden Problemen, wie zum Beispiel die numerische Abbildung unendlicher Integrationsgrenzen bei der Berechnung der Schlitzverschmierung oder das Auffinden von Rahmenbedingungen für die erfolgreiche Faltung der Rockingkurve mit einer beliebigen Streukurve, werden präsentiert. Um die Berechnungszeiten der Streudaten auf ein Minimum zu reduzieren, wurde die Programmierung der Berechnungsmethoden speziell auf die Unterstützung von Multi-Core Prozessoren abgestimmt. Zusätzlich wird eine Methode vorgestellt, wie mit Hilfe von applikationsspezifischen, zweidimensionalen Wave-Objekten (Matrixwaves) und Multithreading mehrdimensionale Integrale effizient berechnet werden können. Im letzten Teil der Arbeit werden mit Hilfe der fertiggestellten Applikation ausgewählte Fragestellungen behandelt, die im Rahmen der Analyse von realen Streudaten auftreten. Dadurch können einige interessante Antworten abgeleitet werden, die erkennen lassen, wie komplex das Zusammenspiel der mannigfaltigen Parameter in Bezug auf die Zusammensetzung der Streukurve ist.

Vielseitige Anwendungsmöglichkeiten bei gleichzeitig einfacher Bedienung machten die Applikation einerseits zu einer wichtigen Komponente dieser Arbeit, andererseits unterstreichen sie aber auch ihr Potential, komplementär in der Streudatenanalyse oder optional als unterstützendes Werkzeug im Lehrbetrieb eingesetzt zu werden.

Abstract

The present work from the area of neutron physics is concerned with a task regarding magnetic ultra-small angle neutron scattering (USANSPOL). It is shown how generation and subsequent analysis of numerical calculated (synthetic) scattering data allows one to draw inferences that could be helpful in analysis of actual scattering data. Central aspect is the development of an IGOR Pro based “USANS simulation application”, which utilizes a simple user interface to generate scattering graphs of various parameter settings rapidly. An introduction into the topic of neutron scattering in general is followed by a mathematical discourse about the theoretical principles, which is used as a foundation for the application`s core components later on. The main part of the work documents the approach on how scattering data synthesis is being realized: First off, the software and its main concepts are presented, and then all steps regarding the development of the USANS simulation application are described in detail. For every section, the underlying mathematical concept and the programmatic implementation is presented, handling of the user interface is explained and finally an exemplary output of the result is shown. Solutions for a variety of occurring problems like on how to handle infinite integration limits during the numerical calculation of the slitheight-smearing or finding the appropriate parameters to successfully convolve the rocking curve with an arbitrary scattering curve are being discussed. To minimize the time required for calculating scattering data, the programming of calculation methods is focused on supporting multi-core processors systems. Furthermore a very efficient method for computing multidimensional integrals by utilizing two-dimensional wave-objects (matrix waves) in multithreaded wave assignments is being introduced. In the final part of this work, selected issues that arise from analysis of real scattering data are being discussed. Several interesting answers can be derived, which, on the other hand, show how complex the interactions between the various parameters are in relation to the composition of the scattering curve.

Miscellaneous applications coupled with simple usability made this application an important component of this work, and also underline its potential to be utilized in either complementary scattering data analysis or as an optional tool in teaching.

Inhaltsverzeichnis

1 Einleitung

1.1 Allgemeines zu SANS/USANS/USANSPOL

Die Kleinwinkelneutronenstreuung (SANS – „Small Angle Neutron Scattering“) ist eine physikalische Methode zur Untersuchung von Materiestrukturen verschiedenster Substanzen mittels elastischer Neutronenstreuung. Im Unterschied zu vergleichbaren analytischen Verfahren die Röntgenstrahlen, Synchrotronstrahlen, Elektronen oder Licht als Untersuchungsmedium einsetzen, zeichnen sich Neutronen vor allem durch ihre speziellen physikalischen Eigenschaften aus. Die davon grundlegendste ist ihre elektrostatische Neutralität, sie erlaubt es den Teilchen, verhältnismäßig tief in das zu untersuchende Probenmaterial einzudringen, wodurch die Bestimmung von Strukturparametern nicht nur auf oberflächennahe Bereiche beschränkt ist. Auf elektromagnetischer Ebene kann in Hinblick auf die Wechselwirkung des magnetischen Moments eine sehr leistungsfähige Methode zur Erforschung von magnetischen Strukturen in Festkörpern bereitgestellt werden. Ein weiteres Unterscheidungsmerkmal zu anderen Streumethoden, die z.B. vorrangig von der Elektronenkonfiguration abhängig sind, ist die Möglichkeit, einzelne Isotope durch ihr unterschiedliches nukleares Streuverhalten zu identifizieren. Aufgrund der qualitativ und quantitativ gegebenen Verfügbarkeit von Neutronenquellen weltweit und der ständigen Weiterentwicklung der experimentellen Infrastruktur sowie der Neutronenoptik hat sich die Kleinwinkelneutronenstreuung seit längerem erfolgreich in Anwendungsgebieten wie Festkörperphysik, Biophysik, Molekularbiologie und Metallurgie etabliert.

Die Bezeichnung Kleinwinkelstreuung stammt von Beugungsbildern, die unter sehr kleinen Streuwinkeln (wenige Grad oder Bogensekunden) entstehen, wenn im Experiment ein Neutronenstrahl mit einer Wellenlänge von einigen Angström an der inneren Struktur einer Probe elastisch gestreut wird. Entgegen der namentlich naheliegenden Erwartung, wird im Bereich SANS zur Charakterisierung einer Messung aber nicht der Streuwinkel, sondern der instrumentenunabhängige Impulsübertrag oder Streuvektor, der durch die Beziehung $q = \frac{4\pi}{\lambda} \sin\left(\frac{\theta}{2}\right)$ definiert ist, angegeben. Im Vergleich zu herkömmlichen SANS Instrumenten kann bei USANS („Ultra Small Angle Neutron Scattering“) durch ein grundlegend unterschiedliches Instrumentenkonzept ein deutlich kleinerer

q-Bereich erreicht werden. Diese Entwicklung in Richtung noch kleinerer Streuwinkel bzw. *q*-Werte wurde in erster Linie durch den Einsatz von thermischen Neutronen und perfekten Einkristallen möglich. Durch die Kombination von SANS und USANS Methoden, die jeweils gewisse q-Bereiche abdecken, erreicht man somit eine nahtlose Auflösung der Probenstruktur im Ortsraum zwischen 1 nm und 30 µm.

Wie schon eingangs erwähnt kann die USANS Technik dafür verwendet werden, magnetische Mikrostrukturen von kondensierter Materie analytisch zu erfassen. Die als USANSPOL („Ultra Small Angle Neutron Scattering Polarized“) bezeichnete Methode unterscheidet sich im Wesentlichen durch das Einbringen von Magnetprismen in den Strahlengang zwischen Monochromator und Probe. Eine durch diese Architektur erreichte Aufspaltung des Neutronenstrahls in eine Spin-up und eine Spin-down Komponente sowie die, in weiterer Folge differenziert beobachtbare, Spin-abhängige Wechselwirkung mit dem Probenmaterial, bildet die Grundlage für dieses Verfahren. USANSPOL ist zum Zeitpunkt der Arbeit eine der aktuellsten Weiterentwicklungen auf dem Gebiet der Neutronenstreuung; Datenmodellierung, Analytik und technisches Setup befinden sich derzeit aber noch im Entwicklungsstadium.

1.2 Forschungsaktivitäten und Motivation

Das Atominstitut der TU Wien führt USANS Experimente an zwei Standorten durch: Intern am 250kW TRIGA Mark II Forschungsreaktor in Wien und extern am 58 MW Hochflussreaktor des Instituts Laue-Langevin (ILL) in Grenoble, Frankreich. Die Nutzung der externen Einrichtung hat vor Allem aus technischer Sicht einen guten Grund: Die am ILL erzeugten Neutronenflussdichten zählen zu den höchsten verfügbaren weltweit - ein Umstand der für Experimente von entscheidender Bedeutung ist. In diesem Zusammenhang werden am Atominstitut der TU Wien in regelmäßigen Abständen Experimente vorbereitet, die dann zu genau definierten Zeiträumen am ILL durchgeführt werden. Im August/September 2009 wurde am neutronenoptischen Instrument S18 eine USANSPOL Experimentserie mit einem neuartigen magnetischen Material auf Eisen-Bor (FeB) bzw. Eisen-Gallium (FeGa)-Basis ausgeführt. Zielsetzung war die Untersuchung der magnetischen Mikrostruktur des in Bandform vorliegenden Probenmaterials, das wegen seiner außergewöhnlichen magnetischen Eigenschaften (z.B. Magnetostriktion) von technischem Interesse war. Das Experiment wurde mit einigen Zwischenfällen, wie einem überraschenden Stromausfall, wodurch wichtige Strahlzeit verloren ging, beendet und die Messergebnisse der Datenauswertung zugeführt. Das Ergebnis einer solchen Datenauswertung wird in Bezug auf eine ähnliche, aber zeitlich aktuellere USANSPOL-Messung [1] in Abbildung 1.1 veranschaulicht.

Um die Interpretation der dabei gemessenen Daten qualitativ zu verbessern und ein besseres Verständnis für die Entstehung der gemessenen Streukurven zu bekommen, wurde die Idee geboren, den gesamten vorhandenen mathematischen Formalismus der hinter der Methode USANSPOL steht, numerisch zu simulieren und die Ergebnisse anschließend mit den Messdaten zu vergleichen. Der Gegenstand dieser Arbeit umfasst dementsprechend die Erarbeitung des mathematischen Formalismus, die darauf aufbauende numerische Simulation und eine Diskussion der resultierenden Ergebnisse.

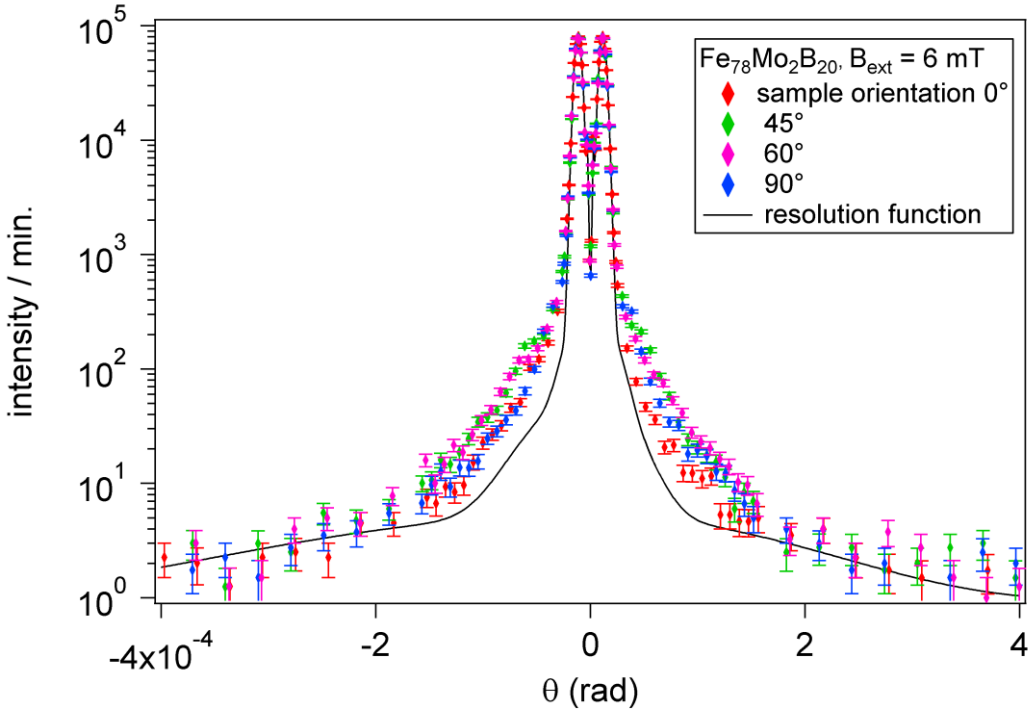

Abbildung 1.1: Ergebnis einer USANSPOL-Messung, die 2011 an einem amorphen, weichmagnetischen FeMoB-Band [1] durchgeführt wurde. Die einzelnen Punkte der Messreihe, die verschiedene Probenausrichtungen umfasst, wurden farblich gekennzeichnet und sind mit Fehlerbalken versehen. Die Auflösungsfunktion des Instruments wurde als Linie (schwarz) aufgetragen.

2 Grundlagen

2.1 Historisches zum Neutron

Das Neutron wurde erstmals um 1920 von einem der bedeutendsten Experimentalphysiker, Sir Ernest Rutherford, theoretisch postuliert. Darauf folgend wurde im Jahre 1930 von Walter Bothe und Herbert Becker beim Beschuss von Beryllium mit Polonium-Alphateilchen eine neue bisher unbekannte Form von Strahlung gefunden. Die beobachtete Kernreaktion lautet:

$$ {}^{9}_{4}Be + {}^{4}_{2}\alpha \rightarrow {}^{12}_{6}C + {}^{1}_{0}n \qquad (2.1) $$

Man vermutete zunächst eine energiereiche Form von Gammastrahlung, als nach einer genaueren Analyse immer mehr Zweifel an dieser Theorie aufkamen, nannte man die Strahlung zwischenzeitlich „Beryllium-Strahlung“. Im folgenden Jahr, 1931, führte das Ehepaar Joliot-Curie weitere Experimente mit der neuen Strahlungsform durch. Sie konnten mit Hilfe von Ionisationskammern weitere detaillierte Versuchsergebnisse liefern, die einen wichtigen Beitrag zur Erforschung der Natur der unbekannten Strahlung darstellten. Mit diesen Ergebnissen in der Hand wurde Sir James Chadwick – einem Schüler Rutherfords – bewusst, dass es sich bei der „Beryllium-Strahlung“ um jene Teilchenstrahlung handeln musste, die sein Lehrmeister schon 12 Jahre zuvor postuliert hatte. Er wiederholte im Jahre 1932 alle Experimente von Joliot-Curie und konnte nachweisen, dass es sich bei der neuen Strahlungsform nicht um Gammastrahlen, sondern um eine Teilchenstrahlung, bestehend aus neutralen Teilchen mit einer Masse ähnlich jener des schon bekannten Protons, handelte. Die elektrisch ungeladenen Teilchen nannte er „Neutronen“, eine Komposition der Wörter „neutral“ und dem für subatomare Partikel üblicherweise als Suffix verwendeten Wort „-on“.

Für diese Entdeckung erhielt Sir Chadwick im Jahre 1932 die Hughes Medal of the Royal Society und in weiterer Folge den Nobelpreis für Physik im Jahre 1935 [2].

2.2 Physikalische Eigenschaften des Neutrons

Das Neutron ist ein subatomares Hadron mit dem Formelzeichen *n* oder n^0. Im ungebundenen Zustand hat es eine Ruhemasse von 1,674 927 351(74) · 10^{-27} kg oder, in atomaren Einheiten ausgedrückt 1,008 664 916 00(43) *u*. Damit ist seine Ruhemasse um 0,14% größer als die des freien Protons. Wie sein Name schon impliziert, trägt es keine elektrische Ladung; es besitzt einen Spin von 1/2, zählt damit zu den Fermionen und hat, bedingt durch seinem Spin, ein magnetisches Moment von −0,966 236 47(23) · 10^{-26} J/T. Das Neutron ist ein zusammengesetztes Objekt, es besteht aus 3 Elementarteilchen, zwei Down-Quarks und einem Up-Quark (Formel: udd), und gehört damit zur Familie der Baryonen. Aufgrund dieser inneren Struktur ist es räumlich ausgedehnt und hat einen Durchmesser von ~1,7 fm. Alle Angaben in diesem Abschnitt wurden den CODATA Recommended Values [3] entnommen.

2.2.1 Elementare Wechselwirkungen

Neutronen unterliegen allen vier in der Physik bekannten Grundkräften bzw. fundamentalen Wechselwirkungen [2]:

Der starken Wechselwirkung als Grundlage für Neutron-Kern-Wechselwirkungen. Die damit verbundene Ausbildung eines neutronenoptischen Potentials ist auf diese Weise für die beobachtbaren Streuphänomene verantwortlich.

Der schwachen Wechselwirkung, die bewirkt, dass das freie Neutron instabil ist und über den Prozess des Betazerfalls mit einer mittleren Lebensdauer von τ=880,1 ± 1,1s [4] in ein Proton, ein Elektron und ein Antineutrino zerfällt:

$$n \rightarrow p^+ + e^- + \overline{\nu}_e \qquad (2.2)$$

Der elektromagnetischen Wechselwirkung; das Neutron ist elektrisch ungeladen und unterliegt daher nicht den elektrostatischen Prozessen wie Anziehung oder Abstoßung, wohl aber interagiert es auf elektromagnetischer Ebene über sein vorhandenes magnetisches Moment. Das daraus resultierende Potential ist die Grundlage für magnetische Streuphänomene.

Als Objekt mit endlicher Ruhemasse unterliegen sie natürlich auch der Gravitation.

2.2.2 Wechselwirkungen mit Materie

Bei der Wechselwirkung von Neutronen mit Materie kommt es zu verschiedenen Prozessen, die abhängig von der auftretenden Neutronenenergie unterschiedlich hohe Eintrittswahrscheinlich-

keiten haben. Ohne näher auf diese Abhängigkeit einzugehen, finden typischerweise folgende Prozesse statt:

- Elastische Streuung an Atomkernen
- Inelastische Streuung an Atomkernen - es kommt zur einer Anregung des Kerns, der durch Emission von Gammastrahlung wieder in den Grundzustand zurückfällt
- Neutroneneinfang, Isotopenbildung
- Kernreaktionen – Kernspaltung, (n,p) - , (n,α) -, (n,2n) - Reaktionen

2.2.3 Klassifizierung

Wie schon im vorigen Kapitel erwähnt, verhalten sich Neutronen in Bezug auf ihre Wechselwirkungen stark energieabhängig. Daher erscheint eine Gruppierung in Energiebandbreiten naheliegend; folgende Einteilung hat sich herauskristallisiert [2]:

Neutronentyp	Kinetische Energie[eV]
Ultrakalte Neutronen	<0.2meV
Kalte Neutronen	<2meV
Thermische Neutronen	<100meV
Epithermische Neutronen	<1ev
Mittelschnelle Neutronen	0.5ev-10keV
Schnelle Neutronen	10keV-20keV
Relativistische Neutronen	>20keV

Die „thermische“ Nomenklatur bezieht sich in diesem Bereich auf das Energiespektrum der Neutronen, deren kinetische Energieverteilung mit der Maxwell-Bolzmannverteilung eines Mediums bei Raumtemperatur vergleichbar ist.

2.2.4 Neutronen als Kernbestandteile

Alle bekannten Atomkerne, mit Ausnahme des Wasserstoffisotops Protium ^{1}H, bestehen sowohl aus Protonen und Neutronen. Beide Kernbestandteile werden als Nukleonen bezeichnet [2]. Das Verhältnis zwischen Protonen und Neutronen ist für kleine Kerne etwa gleich, mit zunehmender

Massenzahl steigt die Anzahl der Neutronen jedoch überproportional an, sie kann bei schweren Kernen fast das 1,5-fache der Protonenzahl betragen. In Atomkernen gebundene Neutronen sind energetisch stabil, wenn die Kerne an die sie gebunden sind ebenfalls stabil sind. Ist ein Neutron an einen instabilen Kern gebunden, besteht genauso wie im Fall eines freien Neutrons, die Möglichkeit eines Betazerfalls. Weist ein chemisches Element mit fester Protonenzahl unterschiedliche Neutronenzahlen auf, so spricht man von einem Isotop.

2.2.5 Neutronenquellen in der Technik

Neutronenquellen dienen zur Herstellung von freien Neutronen für Forschung und Anwendungen der Neutronenphysik [5], [6]. Meist werden primärseitig energiereiche Neutronen erzeugt, die anschließend durch geeignete Moderation in die gewünschte Energiebandbreite gebracht werden. Die wichtigsten Kennzahlen sind hierbei die Quellstärke, also die von der Quelle abgegebenen Neutronen pro Zeiteinheit sowie die Neutronenflussdichte, die unmittelbar für das Experiment zur Verfügung steht. Technisch werden folgende Neutronenquellen eingesetzt:

- Radioaktive Neutronenquellen – Alpha-Beryllium, Gamma-Beryllium und Spontanspaltungsquellen
- Kernreaktoren
- Teilchenbeschleuniger - (p,n) und (d,n) Reaktionen, Spallationsquellen sowie Neutronenerzeugung durch Elektronen-Bremsstrahlung
- Pyroelektrische Fusion – Durch Pyroelektrische Kristalle induzierte (d,n)-Kernreaktion
- Farnsworth-Hirsch Fusor – Kernfusion durch elektrostatischen Plasmaeinschluss

2.2.6 Argumente für den Einsatz von Neutronen

Neutronen eignen sich für Festkörper- und Oberflächenanalysen in kondensierter Materie und ermöglichen die Untersuchung von Probenstruktur und der in ihr vorhandenen Dynamik:

- Neutronen wechselwirken mit Materie über kurzreichweitige Kernkräfte, sie sind sehr durchdringend und verursachen im Probenmaterial nur geringe Erwärmung.
- Die Wellenlänge der Neutronen ist vergleichbar mit atomaren Größenordnungen und den dazwischenliegenden Abständen.
- Neutronenenergien liegen im Bereich normaler Schwingungsmoden in Festkörpern (Phononen und Magnonen).
- Niederenergetische, zerstörungsfreie und nicht-ionisierende Analysemethode.
- Erfassung von magnetischen Materialeigenschaften.
- Isotope können bei nuklearer Streuung differenziert werden.

2.3 Grundgleichungen

Redet man im klassischen physikalischen Sinne über Neutronen, so werden diese als massebehaftete Teilchen betrachtet, die den Gesetzen der ebenfalls klassischen Mechanik gehorchen. Seit der Entdeckung quantenphysikalischer Phänomene Anfang des 20. Jahrhunderts weiß man, dass sich Materie nicht nur eindimensional über seinen Teilchencharakter beschreiben lässt, sondern, je nach gewähltem Experimentaufbau, auch einen ausgeprägten Wellencharakter aufweist. Diese Tatsache wird in der Quantenphysik als Welle-Teilchen Dualismus bezeichnet. Der De-Broglie Gleichung zufolge können Neutronen als Materiewellen der Wellenlänge λ dargestellt werden:

$$\lambda = \frac{h}{p} = \frac{h}{mv} \qquad (2.3)$$

h bezeichnet dabei die zentrale quantenmechanische Konstante, das Plancksche Wirkungsquantum, $p = mv$ ist der Impuls des Teilchens. Gleichung (2.3) ist nur im nicht-relativistischen Bereich gültig, was aber für die in dieser Arbeit betrachteten, kalten und thermischen Neutronen mit einem Geschwindigkeitsverhältnis $\frac{v}{c} < 10^{-5}$ zutreffend ist. Die Beziehungen zwischen kinetischer Energie, Wellenlänge und Temperatur können wie folgt angegeben werden:

$$E = \frac{mv^2}{2} = \frac{\hbar^2 k^2}{2m} = \frac{h^2}{2m\lambda^2} = k_B T \qquad (2.4)$$

Um in der Praxis die Zusammenhänge zwischen den verwendeten Grundgrößen einfach darzustellen kann man nach [7] weitere Relationen für die Energie angeben:

$$E[meV] = 2.0723k^2 = \frac{81.81}{\lambda^2} = 5.2267 \times 10^{-6} v^2 = 0.086173T \qquad (2.5)$$

Bei Raumtemperatur weisen thermische Neutronen folgende nominelle Parameter auf:

$$E = 25meV, \qquad \lambda = 1.8\text{Å}, \qquad v = 2200\,{}^{m}/_{S} \qquad (2.6)$$

3 Theorie der Kleinwinkelneutronenstreuung

Die folgende Abhandlung orientiert sich im Wesentlichen an den theoretischen Abschnitten aus [8], [9] und [10], wobei versucht wurde, die ideale Schnittmenge zwischen diesen Arbeiten zu finden.

3.1 Allgemeine Streutheorie

3.1.1 Streuamplitude, Schrödingergleichung und Bornsche Näherung

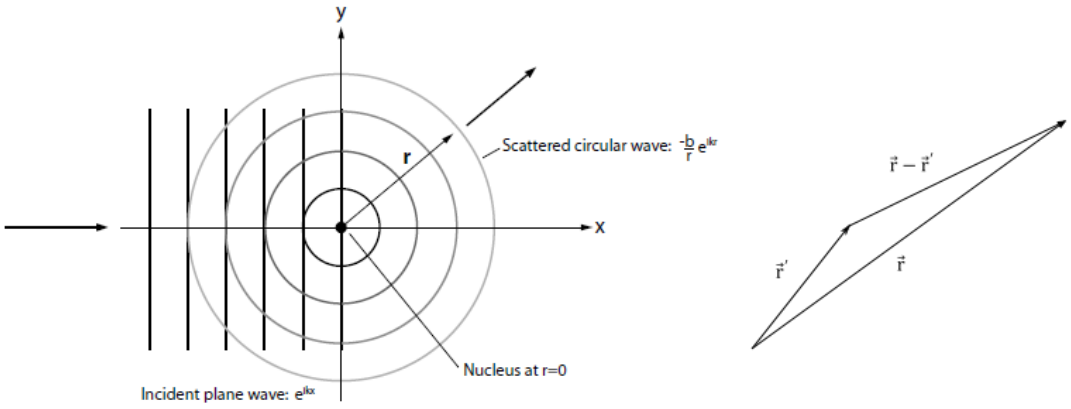

Abbildung 3.1: Schematische Darstellung des Streuprozesses. Rechte Seite: einfallende ebene Welle in Richtung positiver x-Achse und daraus resultierend die am Atomkern gestreute sphärische Welle. Linke Seite: Definition der Abstandsvektoren. Die Graphik wurde Unterlage [11] entnommen.

Im herkömmlichen Sinne werden die Berechnungsmethoden der Quantenmechanik dazu verwendet, Bindungszustände für verschiedene Konfigurationen auf atomarer Ebene zu ermitteln. Obwohl die Problemstellung der Streuung keine Bindungszustände beinhaltet, kommen aufgrund der Wellennatur der betrachteten Teilchen dennoch die gleichen mathematischen Werkzeuge zum Einsatz. Im Falle der Berechnung der Streuamplitude, die hier als erste Größe ermittelt werden soll, ist das die Lösung der zeitunabhängigen Schrödingergleichung $H\psi = E\psi$.

Es sei $H_i = -\frac{\hbar^2}{2m}\nabla^2$ der Hamiltonoperator des einfallenden Neutrons, $E_i = \frac{\hbar^2 k_i^2}{2m}$ seine kinetische Energie, k_i die einfallende Wellenzahl und Ψ_i seine Eigenwertfunktionen. Die homogene Differentialgleichung lautet:

$$(H_i - E_i)\Psi_{\mathrm{i}}(\vec{\mathrm{r}}) = -\frac{\hbar^2}{2\mathrm{m}}\left(\nabla^2 + \mathrm{k}_{\mathrm{i}}^2\right)\Psi_{\mathrm{i}}(\vec{\mathrm{r}}) = 0 \qquad (3.1)$$

Ihre Lösung hat die Form einer ebenen Welle mit der Funktion $\Psi_{\mathrm{i}}(\vec{\mathrm{r}}) = \mathrm{e}^{\mathrm{i}\vec{\mathrm{k}}_{\mathrm{i}}\vec{\mathrm{r}}}$. Die vollständige Differentialgleichung mit Index *s* für die gestreuten Zustände und *V* als Zentralpotential zwischen Neutron und Kern lautet:

$$-\frac{\hbar^2}{2\mathrm{m}}(\nabla^2 + \mathrm{k}_{\mathrm{s}}^2)\Psi(\vec{\mathrm{r}}) = -\mathrm{V}(\vec{\mathrm{r}})\Psi(\vec{\mathrm{r}}) \qquad (3.2)$$

Lösung von (3.2) hat folgende Form:

$$\Psi(\vec{\mathrm{r}}) = \Psi_{\mathrm{i}}(\vec{\mathrm{r}}) + \left(\frac{\mathrm{m}}{2\pi\hbar^2}\right)\int \mathrm{d}\vec{\mathrm{r}}'\mathrm{G}(\vec{\mathrm{r}} - \vec{\mathrm{r}}')\,\mathrm{V}(\vec{\mathrm{r}}')\Psi(\vec{\mathrm{r}}') \qquad (3.3)$$

Dabei wird mit $\mathrm{G}(\vec{\mathrm{r}} - \vec{\mathrm{r}}')$ eine Greensche Funktion eingesetzt die mit k_s, der gestreuten Wellenzahl folgender Differentialgleichung genügt:

$$(H - E_s)G(\vec{\mathrm{r}}) = -\frac{\hbar^2}{2\mathrm{m}}(\nabla^2 + \mathrm{k}_{\mathrm{s}}^2)\mathrm{G}(\vec{\mathrm{r}}) = \delta(\vec{\mathrm{r}}) \qquad (3.4)$$

Die Lösung dieser Gleichung entspricht einer sphärischen ausgehenden Welle der Form $\mathrm{G}(\vec{\mathrm{r}}) = \frac{e^{ik_s r}}{r}$.

Im Allgemeinen wird eine Schrödingergleichung wie in (3.2) durch eine Integralgleichung (3.3) gelöst, die ihre Lösung durch iterative Entwicklung findet:

$$\begin{aligned}\Psi(\vec{\mathrm{r}}) = \Psi_{\mathrm{i}}(\vec{\mathrm{r}}) &+ \left(\frac{\mathrm{m}}{2\pi\hbar^2}\right)\int \mathrm{d}\vec{\mathrm{r}}'\mathrm{G}(\vec{\mathrm{r}} - \vec{\mathrm{r}}')\,\mathrm{V}(\vec{\mathrm{r}}')\Psi_{\mathrm{i}}(\vec{\mathrm{r}}') \\ &+ \left(\frac{\mathrm{m}}{2\pi\hbar^2}\right)^2 \int \mathrm{d}\vec{\mathrm{r}}'\mathrm{G}(\vec{\mathrm{r}} - \vec{\mathrm{r}}')\,\mathrm{V}(\vec{\mathrm{r}}')\Psi_{\mathrm{i}}(\vec{\mathrm{r}}') \int \mathrm{d}\vec{\mathrm{r}}''\mathrm{G}(\vec{\mathrm{r}} - \vec{\mathrm{r}}'')\,\Psi_{\mathrm{i}}(\vec{\mathrm{r}}'') + \cdots\end{aligned} \qquad (3.5)$$

Beim Vernachlässigen der Terme höherer Ordnung spricht man von der Bornschen Näherung, die für effektive Streupotentiale, die klein im Vergleich zur Energie des einfallenden Wellenfeldes sind, und für Vernachlässigung von Mehrfachstreuung gültig ist. Da die gestreuten Teilchen in

großer Entfernung zum Streuzentrum beobachtet werden, ist es zulässig, eine Fernfeld-Näherung, die sogenannte Fraunhofer-Näherung, anzusetzen: Es ist $r \gg r'$ und man kann $|\vec{\mathrm{r}} - \vec{\mathrm{r}}'| \cong r - \frac{\vec{\mathrm{r}}\cdot\vec{\mathrm{r}}'}{r}$ annähern. Die Greensche Funktion hat dadurch folgende Form:

$$\mathrm{G}(|\vec{\mathrm{r}} - \vec{\mathrm{r}}'|) = \frac{e^{ik_s r}}{r} e^{-\frac{ik_s \vec{\mathrm{r}}\cdot\vec{\mathrm{r}}'}{r}} = \frac{e^{ik_s r}}{r} e^{-i\vec{k_s}\cdot\vec{\mathrm{r}}'} \tag{3.6}$$

Zusammengefasst und in die Bornsche Näherung eingesetzt ergibt sich:

$$\begin{aligned} \Psi(\vec{\mathrm{r}}) &= \mathrm{e}^{\mathrm{i}\vec{\mathrm{k}}_\mathrm{i}\vec{\mathrm{r}}} + \frac{e^{ik_s r}}{r}\left(\frac{\mathrm{m}}{2\pi\hbar^2}\right)\int \mathrm{d}\vec{\mathrm{r}}' e^{-i\vec{k_s}\cdot\vec{\mathrm{r}}'}\, \mathrm{V}(\vec{\mathrm{r}}')\mathrm{e}^{\mathrm{i}\vec{\mathrm{k}}_\mathrm{i}\vec{\mathrm{r}}'} \\ &= \mathrm{e}^{\mathrm{i}\vec{\mathrm{k}}_\mathrm{i}\vec{\mathrm{r}}} + \frac{e^{ik_s r}}{r} f(\theta) \end{aligned} \tag{3.7}$$

Durch Einsetzen des Streuvektors $\vec{Q} = \vec{k}_s - \vec{k}_i$ wird schließlich die Streuamplitude definiert:

$$f(\theta) = \left(\frac{\mathrm{m}}{2\pi\hbar^2}\right)\int \mathrm{d}\vec{\mathrm{r}}' e^{-i\vec{Q}\cdot\vec{\mathrm{r}}'}\, \mathrm{V}(\vec{\mathrm{r}}') \tag{3.8}$$

Die Streuamplitude ist somit proportional zur Fourier-Transformierten des Wechselwirkungspotentials zwischen Neutron und Kern und hat die Dimension einer Länge. Die Bornsche Näherung ist, mit Ausnahme von Reflexion, für alle Streuprozesse an denen thermischen und kalten Neutronen beteiligt sind, gültig. Für die Berechnung der Reflexion wären in der iterativen Entwicklung Terme höherer Ordnung notwendig.

Zusammengefasst sollte erwähnt werden, dass die Lösung der Schrödingergleichung eine Summe aus einfallender ebener Welle und gestreuter sphärischer Welle multipliziert mit der Streuamplitude ist.

3.1.2 Wirkungsquerschnitt

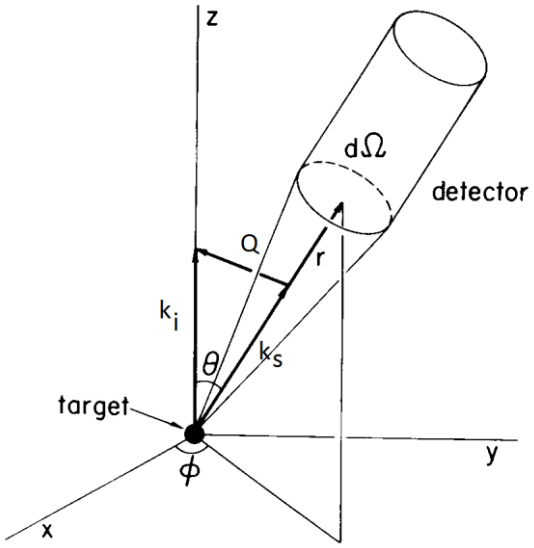

Abbildung 3.2: Darstellung der Streugeometrie nach [12], der Zusammenhang zwischen einfallenden Neutronen mit Wellenzahl $\vec{k}_i$ und innerhalb eines Winkelelements $d\Omega$ gestreuten Neutronen mit Wellenzahl $\vec{k}_s$ wird veranschaulicht. Der Streuvektor *Q* resultiert aus der Differenz zwischen einfallendem und gestreutem Wellenvektor.

Der differentielle Wirkungsquerschnitt erfasst die Anzahl der in das differentielle Raumwinkelelement gestreuten Teilchen bezogen auf den Fluss der einfallenden Teilchen. Ausgangspunkt für die Berechnung des Wirkungsquerschnittes ist die Teilchenstromdichte $\vec{J}$ für einen Zustand der Wellenfunktion $\Psi(\vec{r})$. Die einfallende Teilchenstromdichte in der Einheit [Neutronen/cm^{2}s] ist gegeben durch:

$$\vec{J}_i = \frac{i\hbar}{2m}\left(\Psi_i \vec{\nabla}\Psi_i^* - \Psi_i^* \vec{\nabla}\Psi_i\right) \qquad (3.9)$$

Durch Anwenden des ∇-Operators auf die Wellenfunktion $\vec{\nabla}\Psi = i\vec{k}_i\Psi$ erhält man $\vec{J}_i = \frac{\hbar\vec{k}_i}{m}$. Durch Definition der Funktion χ

$$\chi = \Psi - \Psi_i = \frac{e^{ik_s r}}{r} f(\theta) \qquad (3.10)$$

ergibt sich analog dazu die Teilchenstromdichte nach der Streuung:

$$\vec{J}_s = \frac{i\hbar}{2m}\left(\chi \vec{\nabla}\chi^* - \chi^* \vec{\nabla}\chi\right) \qquad (3.11)$$

In diesem Fall erhält man durch Anwenden des ∇-Operators für $\vec{J}_s = \frac{\hbar \vec{k}_s}{mR^2}|f(\theta)|^2$. Abschließend wird der differentielle Wirkungsquerschnitt definiert:

$$d\sigma_s = \vec{J} d\vec{S} = \frac{J_s}{J_i} r^2 d\Omega = \frac{k_s}{k_i}|f(\theta)|^2 d\Omega \qquad (3.12)$$

Durch Einsetzen der Streuamplitude aus Gleichung (3.8) erhält man als Endergebnis:

$$\frac{d\sigma_s(\Theta)}{d\Omega} = \frac{k_s}{k_i}\left(\frac{\mathrm{m}}{2\pi\hbar^2}\right)^2 \left| \int \mathrm{d}\vec{\mathrm{r}}' e^{-i\vec{Q}\cdot\vec{\mathrm{r}}'} \mathrm{V}(\vec{\mathrm{r}}') \right|^2 \qquad (3.13)$$

Analog zur Definition der Streuamplitude erkennt man, dass der differentielle Wirkungsquerschnitt proportional zum Absolutquadrat der Fouriertransformierten des Potentials ist. Des Weiteren ergibt sich der gesamte Wirkungsquerschnitt durch Integration über den Raumwinkel:

$$\sigma_s = \int \frac{d\sigma_s}{d\Omega} d\Omega \qquad (3.14)$$

Der Wirkungsquerschnitt hat die Dimension einer Fläche und wird in der Einheit 1 barn = 10^{-24}cm^2 angegeben. Seine Form beinhaltet Informationen über die Art und die Struktur des Streumediums. Zu beachten ist, dass dieses Ergebnis nicht für die Fortpflanzungsrichtung (Vorwärtsrichtung) der einfallenden ebenen Welle gültig ist, da es hier aufgrund der gesamten Flusserhaltung zu destruktiven Interferenzerscheinungen kommt.

3.2 Neutronenstreutheorie

Die in den vorigen Abschnitten präsentierten allgemeinen Konzepte sollen nun im Detail auf Neutronen und deren Wechselwirkungen angepasst werden. Wie schon in Abschnitt 2.2.1 beschrieben, stellt die starke Wechselwirkung mit Atomkernen die maßgebliche Größe für die Streuung von Neutronen dar, sie fließt in Form eines Zentralpotentials $V(R)$ in die Berechnung ein, allerdings ist eine vollständige Beschreibung der Kernkraft und des damit anzusetzenden Potentials äußerst komplex. Um diese Schwierigkeit zu umgehen, wurde von Enrico Fermi eine vereinfachte Variante des Kernpotentials in Form eines sogenannten Fermi-Pseudopotentials eingesetzt:

$$V(\vec{r}) = \frac{2\pi\hbar^2}{m} b\delta(\overrightarrow{r}), \qquad (3.15)$$

wobei $\delta(\overrightarrow{r})$ als Diracsche Deltafunktion und b als gebundene Streulänge bezeichnet wird. Die Verwendung eines δ-förmigen Potentials ist insofern zulässig, als die Reichweite der Kernkraft mit $\sim 10^{-15} m$ wesentlich kleiner ist als z.B. die Wellenlänge thermischer Neutronen mit $\sim 10^{-10} m$. Die gebundene Streulänge ist ein Maß für die Stärke der Wechselwirkung zwischen Neutron und Kern, sie hat die Dimension einer Länge und ist positiv für abstoßendes Potential; anziehendes Potential hat negative Werte zur Folge. Wenn die gleichzeitige Wechselwirkung mit mehreren Streuzentren betrachtet wird, so kommt es zu einer mittleren Phasenverschiebung, womit die kohärente Streulänge b_c definiert ist. Geht man zur Streuung an einem makroskopischen Objekt über, so lässt sich das Potential als eine Überlagerung von Einzelpotentialen in der Form

$$V(\vec{r}) = \frac{2\pi\hbar^2}{m} \sum_j b_j \delta(\vec{r} - \vec{r}_j) \qquad (3.16)$$

schreiben.

Ersetzt man für den differentiellen Wirkungsquerschnitt aus Gleichung (3.13) das Potential mit diesem Term und geht zusätzlich von elastischer Streuung aus, was $\vec{k}_s = \vec{k}_i$ bzw. $\frac{k_s}{k_i} = 1$ zur Folge hat, so erhält man den einfach-differentiellen Wirkungsquerschnitt:

$$\begin{aligned} \frac{d\tilde{\sigma}(Q)}{d\Omega} &= \left(\frac{\mathrm{m}}{2\pi\hbar^2}\right)^2 \left| \int \mathrm{d}\vec{\mathrm{r}}' e^{-i\vec{Q}\cdot\vec{\mathrm{r}}'} \mathrm{V}(\vec{\mathrm{r}}') \right|^2 \\ &= \left| \int \sum_{\mathrm{j}} \mathrm{d}\vec{\mathrm{r}}' e^{-i\vec{Q}\cdot\vec{\mathrm{r}}'} b_j \delta(\vec{r} - \vec{r}_j) \right|^2 \\ &= \left| \sum_j b_j e^{-i\vec{Q}\overrightarrow{r_j}} \right|^2 \end{aligned} \qquad (3.17)$$

Die Summation in (3.17) erstreckt sich dabei über alle Atomkerne *j* im makroskopischen Objekt. Bei der Anwendung der Kleinwinkelneutronenstreuung in der Experimentalphysik ist man normalerweise nicht an Streuergebnissen interessiert, die von einzelnen Atomkernen herrühren. Vielmehr will man die Eigenschaften einer makroskopischen Struktur beobachten. Dafür ersetzt man

die individuelle Streulänge jedes Kerns durch eine über ein ausreichend großes makroskopisches Volumen *V(r)* gemittelte Streulängendichte $\rho(\vec{r}) = N(\vec{r})b_c(\vec{r})$:

$$\langle \sum_j b_j \delta(\vec{r} - \vec{r}_j) \rangle_{V(\vec{r})} = N(\vec{r})b_c(\vec{r}) = \rho(\vec{r}), \qquad (3.18)$$

wobei N die Gesamtzahl der Atome ist. Für den einfachen makroskopischen differentiellen Wirkungsquerschnitt folgt dann:

$$\frac{d\tilde{\sigma}(\mathrm{Q})}{d\Omega} = \left| \int_V d\vec{r}\, e^{-i\vec{Q}\cdot\vec{r}'} \rho(\vec{r}) \right|^2 \qquad (3.19)$$

Um den inneren Aufbau der streuenden Struktur detaillierter zu beschreiben, kann die mittlere Streulängendichte in zwei weitere Terme aufgespalten werden:

$$\rho(\vec{r}) = \langle \rho \rangle + D(\vec{r}) \qquad (3.20)$$

Mit der Zerlegung in einen homogenen, strukturlosen Teil $\langle \rho \rangle$ und einen inhomogenen, räumlich strukturierten Anteil $D(\vec{r})$ wird ein wichtiger Parameter der Kleinwinkelstreuung eingeführt: der Streulängendichtekontrast $D(\vec{r})$. In einem Experiment wird ein umso besserer Kontrast zwischen einer homogenen Matrix und eingebetteten Inhomogenitäten erreicht, je größer der Unterschied zwischen den jeweiligen Streulängendichten ist. Bei einem kleinen Streulängendichtekontrast erscheinen die Streubilder „verschmiert“, d.h. sie sind aufgrund ihrer geringen Intensität nur schwer oder undeutlich zu beurteilen.

3.2.1 Zweiphasenmodell

Die praktische Anwendung des zuvor diskutierten Ansatzes findet sich bei der Berechnung des differentiellen Wirkungsquerschnitts für ein allgemeines System, dass, wie in Abbildung 3.3 dargestellt, aus zwei Phasen besteht, wieder.

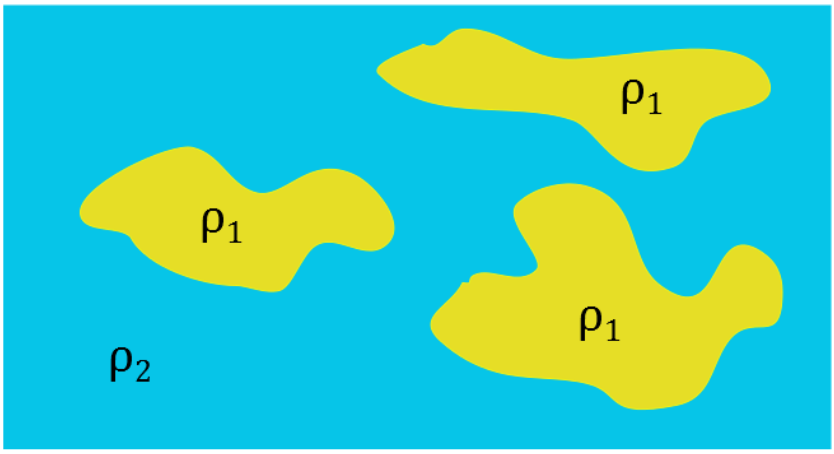

Abbildung 3.3: Schematische Darstellung eines aus zwei Phasen mit den Streulängendichten ρ_1 und ρ_2 bestehenden System

Die beiden Phasen sind durch ihre voneinander unterschiedlichen, aber konstanten Streulängendichten charakterisiert. Dieser Fall tritt relativ häufig auf, z.B. wenn eine metallische Matrix mit unregelmäßigen Einschlüssen oder Ausscheidungen durchsetzt ist. Es seien Volumen und Streulängendichte für dieses System folgendermaßen definiert:

$$V = V_1 + V_2 \tag{3.21}$$

$$\rho(\vec{r}) = \begin{cases} \rho_1 \; in \; V_1 \\ \rho_2 \; in \; V_2 \end{cases} \tag{3.22}$$

Einsetzen in Gleichung (3.19) führt zu:

$$\begin{aligned} \frac{d\tilde{\sigma}(\mathrm{Q})}{d\Omega} &= \left| \int_V d\vec{r}\, e^{-i\vec{Q}\cdot\vec{r}'} \rho(\vec{r}) \right|^2 \\ &= \left| \int_{V_1} d\vec{r}\, e^{-i\vec{Q}\cdot\vec{r}'} \rho_1 + \int_{V-V_1} d\vec{r}\, e^{-i\vec{Q}\cdot\vec{r}'} \rho_2 \right|^2 \\ &= \left| \int_{V_1} d\vec{r}\, e^{-i\vec{Q}\cdot\vec{r}'} (\rho_1 - \rho_2) + \int_V d\vec{r}\, e^{-i\vec{Q}\cdot\vec{r}'} \rho_2 \right|^2 \end{aligned} \tag{3.23}$$

Der zweite Integralterm in (3.23) erstreckt sich über das ganze Volumen des Systems und liefert bei *Q*-Werten ungleich null praktisch keinen Beitrag. Um die Integration wie üblich auf ganz $\mathbb{R}^3$ ausdehnen zu können, wird eine zusätzliche Funktion eingeführt, die nur innerhalb des Streuervolumens V_s einen Wert liefert. Die sogenannte Formfunktion ist definiert als:

$$s(\vec{r}) = \begin{cases} 0, & \vec{r} > V_s \\ 1, & \vec{r} \leq V_s \end{cases} \qquad (3.24)$$

Weiter wird mit $\Delta\rho$ die Streulängendichtedifferenz definiert:

$$\Delta\rho = \rho_1 - \rho_2 \qquad (3.25)$$

Die Streulängendichtedifferenz beschreibt sowohl die Materialeigenschaften wie Dichte und Beschaffenheit, als auch Strahlungseigenschaften wie Streulängendichte des Materials. Damit ergibt sich für den differentiellen Wirkungsquerschnitt eines Zweiphasensystems:

$$\begin{aligned} \frac{d\tilde{\sigma}(\mathrm{Q})}{d\Omega} &= (\Delta\rho)^2 \left| \int d\vec{r}\, e^{-i\vec{Q}\cdot\vec{r}'} s(\vec{r}) \right|^2 \\ &\equiv (\Delta\rho)^2 V^2 |F(Q)|^2 \\ &\equiv (\Delta\rho)^2 V^2 S(Q) \end{aligned} \qquad (3.26)$$

Der Integralterm aus Gleichung (3.26)

$$F(Q) = \frac{1}{V} \int d\vec{r}\, e^{-i\vec{Q}\cdot\vec{r}'} s(\vec{r}) \qquad (3.27)$$

wird als Formfaktor bezeichnet und kann für verschiedenste Teilchengeometrien analytisch berechnet werden. Die häufig verwendete Streufunktion ist definiert durch

$$S(Q) = |F(Q)|^2 \qquad (3.28)$$

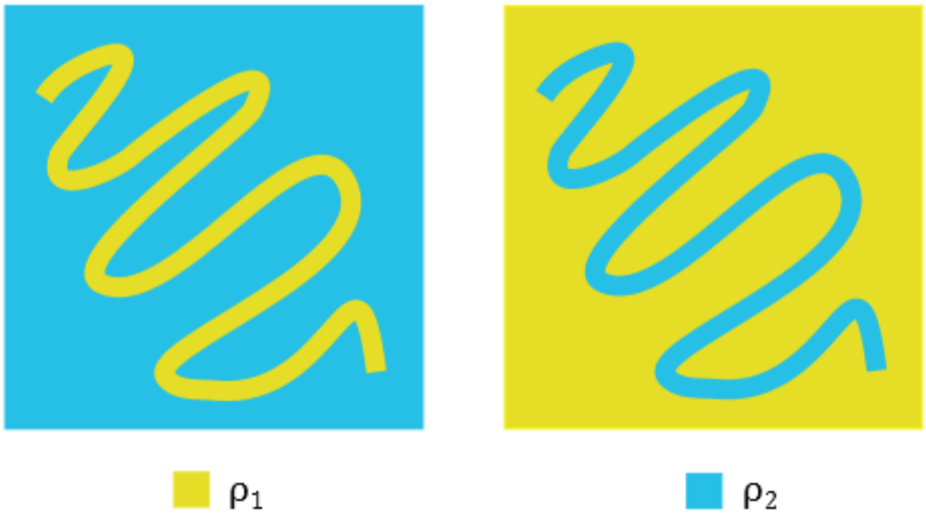

Abbildung 3.4: Zwei strukturell identische Systeme, die sich durch gespiegelten Streulängendichten unterscheiden; Grundlage des Babinetschen Prinzips

Gleichung (3.26) führt zu einer Überlegung, die als Babinetsches Prinzip bezeichnet wird [11]: Zwei strukturell identische Systeme, die sich nur durch gespiegelte Streulängendichten unterscheiden, liefern theoretisch dasselbe Streubild. Eine Erklärung dafür ist der Verlust von Phaseninformation – es kann mittels eines einzelnen Streuexperiments nicht festgestellt werden ob ρ_1 größer ist als ρ_2, bzw. umgekehrt. Beim Durchführen von Kleinwinkelstreuexperimenten sollte deshalb mit einer Kontrastvariation, z.B. durch Zugabe von Deuterium gearbeitet werden, um festzustellen, wie sich das Probenmaterial im Vergleich zu einer bekannten Substanz verhält.

3.2.2 Formfaktoren einfacher Streuer

3.2.2.1 Kugelförmige Teilchen

Die (idealen) streuenden Teilchen werden als kugelförmig, monodispersiv und homogen angenommen, allgemein ist ihr Formfaktor:

$$F(Q) = \frac{1}{V}\int d\vec{r}\, e^{-i\vec{Q}\cdot\vec{r}'} s(\vec{r}) \qquad (3.29)$$

Für einen Kugelradius R ist die entsprechende Formfunktion:

$$s(\vec{r}) = \begin{cases} 0, & \vec{r} > R \\ 1, & \vec{r} \leq R \end{cases} \qquad (3.30)$$

Einsetzen von $s(\vec{r})$ und Transformation in Kugelkoordinaten von Gleichung (3.29) ergibt:

$$F(Q) = \frac{3}{4\pi R^3} \int_0^R \int_0^{2\pi} \int_0^{\pi} e^{-iQr\cos(\vartheta)} r^2 \sin\vartheta \, d\vartheta \, d\varphi \, dr \qquad (3.31)$$

Die Integration über ϑ in (3.31) erfolgt mittels Substitution von $u = Qr\cos\vartheta$ und $\sin\vartheta \, d\vartheta = -\frac{1}{Qr} du$, die Integrationsgrenzen werden mit $0: u = Qr$ und $\pi: u = -Qr$ transformiert:

$$\begin{aligned} &\int_0^{\pi} e^{-iQr\cos(\vartheta)} \sin\vartheta \, d\vartheta = \int_{-Qr}^{Qr} e^{-iu} \frac{1}{Qr} du = \frac{i}{Qr} e^{iu} \Big|_{-Qr}^{Qr} \\ &= \frac{i}{Qr} \frac{2i}{2i} \left(e^{iQr} - e^{-iQr}\right) = \frac{2}{Qr} \sin Qr \end{aligned} \qquad (3.32)$$

Das Integral über die Kugelkoordinate φ ergibt 2π, als Endergebnis für den Formfaktor eines sphärischen Streuers folgt somit:

$$F(Q) = \frac{3}{R^3} \int_0^R r^2 \frac{\sin Qr}{Qr} = 3 \frac{\sin QR - QR\cos QR}{(QR)^3} \qquad (3.33)$$

Abbildung 3.5 stellt die Funktion (3.33) in Form einer Streufunktion (3.28) graphisch dar. In der Literatur wird der Formfaktor oft mit Hilfe der sphärischen Besselfunktion erster Ordnung dargestellt, sie ist definiert durch:

$$j_n(z) = (-1)^n z^n \left(\frac{d}{z\,dz}\right)^n \frac{sin(z)}{z} \qquad (3.34)$$

Sphärische Besselfunktionen können auch durch gewöhnliche Besselfunktionen dargestellt werden:

$$j_v(z) \equiv \sqrt{\frac{\pi}{2z}} J_{v+1/2}(z) \qquad (3.35)$$

Die sphärische Besselfunktion erster Ordnung lautet:

$$j_1(z) = \frac{\sin z}{z^2} - \frac{\cos z}{z} \qquad (3.36)$$

Unter Einbeziehung von (3.36) kann der Formfaktor (3.33) alternativ dargestellt werden durch:

$$F(Q) = \frac{3j_1(QR)}{QR} \qquad (3.37)$$

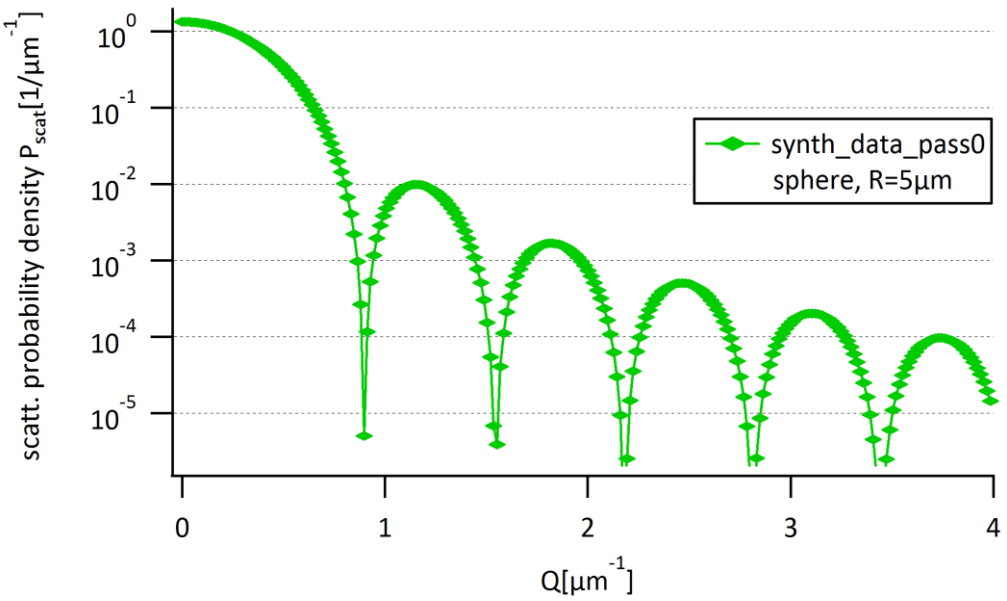

Abbildung 3.5: Berechnete Streufunktion kugelförmiger Streuer mit einem Radius R=5 µm. Die Fläche unter der Kurve wurde auf den Wert 1 normiert, siehe Kapitel 4.2.2.

3.2.2.2 Zylinderförmige Teilchen

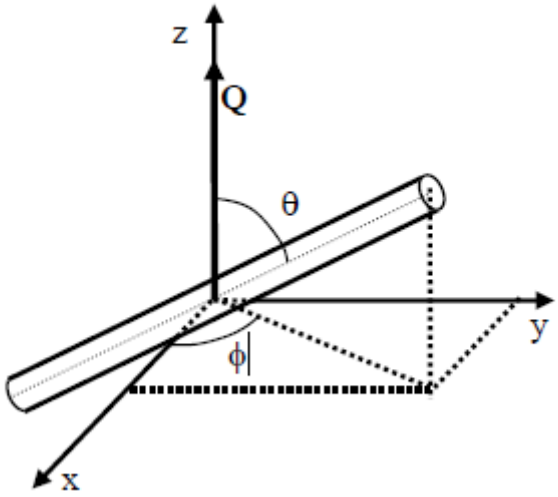

Abbildung 3.6: Definition der Geometrie eines zylinderförmigen Teilchens

Der Formfaktor eines zylinderförmigen Teilchens mit Radius *R* und Länge *L* ist gemäß der Herleitung in [9] gegeben durch:

$$F(Q,\mu) = \left(\frac{sin\left(\frac{Q\mu L}{2}\right)}{\frac{Q\mu L}{2}}\right)\left(\frac{2J_1\left(Q\sqrt{1-\mu^2}R\right)}{Q\sqrt{1-\mu^2}R}\right) = F_z(Q,\mu)F_\perp(Q,\mu) \qquad (3.38)$$

Wobei $\mu = cos(\theta)$ den Neigungswinkel zwischen Zylinderachse und *Q*-Vektor angibt. $F_z(Q,\mu)$ und $F_\perp(Q,\mu)$ sind die jeweiligen Anteile des Formfaktors in longitudinaler Richtung der z-Achse bzw. transversal, also normal zur Zylinderachse. Für ein System zufällig orientierter Zylindergeometrien muss über alle Raumwinkel gemittelt werden:

$$S(Q) = \frac{1}{2}\int_{-1}^{1} d\mu\, |F(Q,\mu)|^2 \qquad (3.39)$$

Die Streufunktion einer richtungsgemittelten Zylindergeometrie (3.39) wird in Abbildung 3.7 graphisch dargestellt. Der Kurvenverlauf ist der sphärischen Geometrie ähnlich.

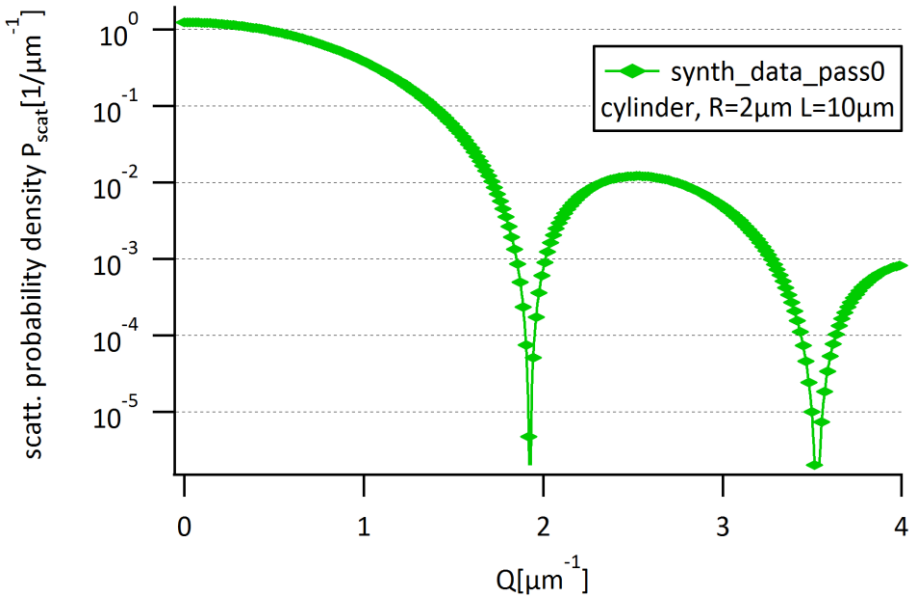

Abbildung 3.7: Berechnete Streufunktion richtungsgemittelter, zylinderförmiger Streuer mit Länge L=10 µm und Radius=2 µm. Die Fläche unter der Kurve wurde auf den Wert 1 normiert, siehe Kapitel 4.2.2.

3.2.2.3 Plättchenförmige Teilchen

Plättchenförmige Teilchen [9] sind ein Spezialfall der zylindrischen Geometrie, ihr Radius R ist groß im Verhältnis zur Dicke (Länge) des Teilchens. Der Formfaktor (3.38) vereinfacht sich durch Ansetzen der Grenzwertnäherung $L \to 0$, die zugehörige richtungsgemittelte Streufunktion wird angegeben mit:

$$S(Q) = \frac{2}{(QR)^2}\left[1 - \frac{J_1(2QR)}{QR}\right] \qquad (3.40)$$

3.2.2.4 Stabförmige Teilchen

Stabförmige Teilchen [9] sind ein Spezialfall der zylindrischen Geometrie, ihre Länge L ist groß im Verhältnis zum Radius R des Teilchens. Der Formfaktor (3.38) vereinfacht sich durch Ansetzen der Grenzwertnäherung $R \to 0$, die richtungsgemittelte Streufunktion wird angegeben mit:

$$S(Q) = \left(\frac{2}{QL}\right) sinc(QL) - \left(\frac{\sin\left(\frac{QL}{2}\right)}{\frac{QL}{2}}\right)^2 \qquad (3.41)$$

3.2.3 Guinier/Porod Näherungen

Guinier [13] und Porod [14] haben für jeweils unterschiedliche Bereiche der Streukurve gezeigt, dass sich bei der Berechnung des Formfaktors praktische Näherungen ansetzen lassen. Im Rahmen einer Reihenentwicklung des Formfaktors für den Bereich $qR \ll 1$ kann gezeigt werden, dass das Quadrat des Formfaktors unabhängig von der Teilchengeometrie ist, vorausgesetzt sie besitzt keine ausgezeichnete Orientierungsrichtung:

$$|F(Q)|^2 = V^2 e^{-\frac{q^2 R_g^2}{3}} \qquad (3.42)$$

Das Absolutquadrat des Formfaktors ist abhängig vom Parameter R_g^2, dem Guinierradius, der wegen seiner mathematischen Struktur auch Trägheitsradius oder Streumassenradius genannt wird. Er ist in seinem Aufbau dem Trägheitsmoment aus der Mechanik ähnlich und beschreibt zusammengesetzte Partikel als eine Summe von N gleichförmigen Elementen, die den Ortsvektoren $\overrightarrow{r_1} \dots \overrightarrow{r_N}$ zugeordnet sind. Der quadratische Trägheitsradius ist dann definiert als der mittlere quadratische Abstand der Elemente vom Schwerpunkt $\overrightarrow{r_s}$ des zusammengesetzten Partikels:

$$R_g^2 = \frac{1}{N}\sum_{i=1}^{N} |\vec{r}_i - \vec{r}_s|^2 \qquad (3.43)$$

Handelt es sich bei den Elementen um eine Massenverteilung mit Massendichte $\rho(\vec{r})$, dann gilt für den Trägheitsradius weiter:

$$R_g^2 = \frac{1}{M}\int d\vec{r}\, \rho(\vec{r}) |\vec{r} - \vec{r}_s|^2 \qquad (3.44)$$

Guinier- oder Trägheitsradien können für verschiedene Teilchengeometrien analytisch ermittelt werden [15]:

$$Kugel: R_g = \sqrt{\frac{3}{5}} R$$

$$Zylinder: R_g = \frac{1}{\sqrt{2}} R$$

$$Plättchen: R_g = \frac{1}{\sqrt{12}} D$$

Ist die Geometrie der Streuer und auch R_g im Gültigkeitsbereich der Guinier-Näherung bekannt, so kann mittels dieser Gleichungen auf die Größe der Streuer geschlossen werden.

Für den entgegengesetzten Fall großer Streuvektoren $qR \gg 2\pi$ kann ein Potenzgesetz abgeleitet werden, welches als Porod-Gesetz bezeichnet wird. Führt man die Reihenentwicklungen für die weiter oben erwähnten Teilchengeometrien durch, so zeigen sich folgende Abhängigkeiten der Potenz von q und der Teilchengeometrie:

$$Kugel: \frac{d\sigma}{d\Omega} \propto \frac{C_p}{q^4}$$

$$Zylinder: \frac{d\sigma}{d\Omega} \propto \frac{C_p}{q^3}$$

$$Plättchen: \frac{d\sigma}{d\Omega} \propto \frac{C_p}{q^2}$$

Außerdem kann für kugelförmige Streuer die Oberfläche der Kugeln aus der Porodkonstante C_p bestimmt werden. Aus der Reihenentwicklung folgt

$$\frac{d\sigma(Q)}{d\Omega} \cong V^2(\Delta\rho)^2 \frac{9}{2} \frac{1}{q^4 R^4} = 2\pi(\Delta\rho)^2 \frac{S}{q^4} \qquad (3.45)$$

mit $S = 4\pi R^2$ als Gesamtoberfläche der Kugeln in der Probe.

Während im Guinierbereich die Möglichkeit besteht die Größe der Streuer zu bestimmen, kann im Porodbereich unmittelbar auf ihre Geometrie geschlossen werden.

3.2.4 Streuung durch magnetische Effekte

Wie schon eingangs erwähnt, ist die mittlere Ladungsdichte des Neutrons innerhalb des Confinement-Radius null, elektrostatische Wechselwirkungen wie Anziehung und Abstoßung mit ladungsbehafteten Objekten finden daher nicht statt. Durch seinen Spin von ½ und dem damit verbundenen magnetischen Moment interagiert das Neutron aber sehr wohl auf elektromagnetischer Ebene. Bei der Streuung von Neutronen in magnetischen Materialien kommt es zu einer Überlagerung der von Atomkernen verursachten nuklearen Streueffekte und der durch Dipol-Dipol Wechselwirkung verursachten magnetischen Streueffekte.

Als Grundlage für die theoretische Behandlung dieses Themas wurde das Skriptum „Solid State Spectroscopy II“ [16] gewählt.

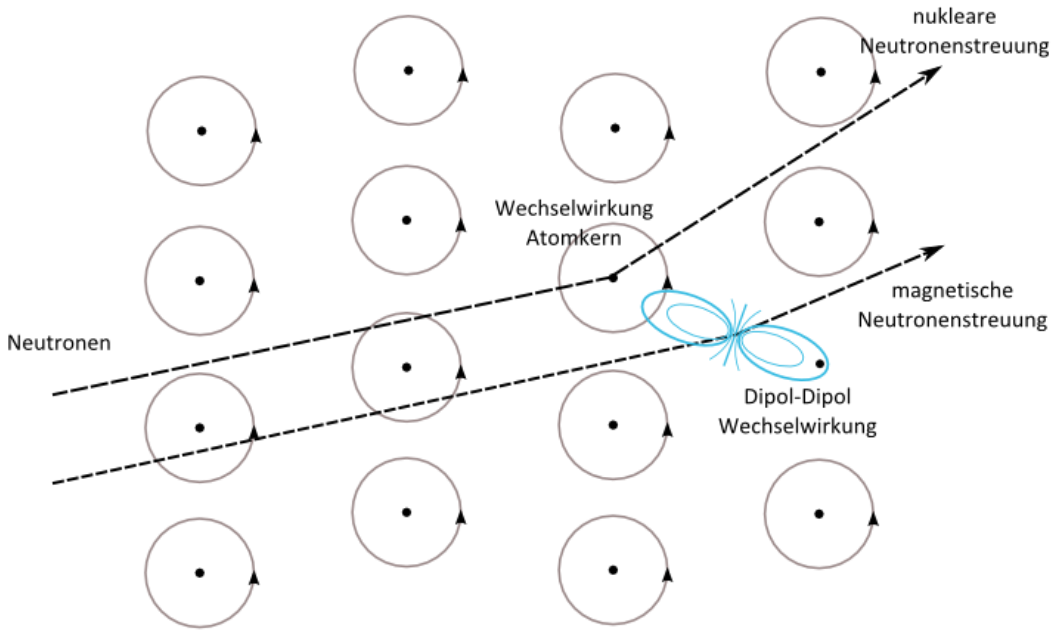

Abbildung 3.8: Schematische Darstellung der nuklearen und magnetischen Wechselwirkung von Neutronen beim Durchtritt durch einem Atomverband

Das Ziel, die magnetische Streuung im Folgenden mathematisch zu beschreiben, wird, wie bei der nuklearen Streuung, die Herleitung des Wirkungsquerschnitts sein. Die magnetischen Momente von Elektron und Neutron sind gegeben durch:

$$\vec{\mu}_e = -2\mu_B \vec{S}_e \qquad \mu_B = \frac{e\hbar}{2m_e} \tag{3.46}$$

$$\vec{\mu}_n = -g_n \mu_N \vec{S}_n = -\gamma \mu_N \vec{\sigma} \qquad \mu_N = \frac{e\hbar}{2m_n} \tag{3.47}$$

mit den Spin ½-Operatoren $\vec{S}_e$ und $\vec{S}_n$, sowie $\vec{\sigma}$, der Pauli-Matrizen für das Neutron, deren Eigenwerte +/-1 sind. Der Ansatz des differentiellen Wirkungsquerschnitts für ein ruhendes Elektron im Ursprung des Koordinatensystems ist ähnlich der nuklearen Streuung, er folgt Fermis goldener Regel aus der quantenmechanischen Störungstheorie. Der Hauptunterschied liegt in der Mitführung des Neutronenspinzustandes $|m\rangle = |\pm 1\rangle$ zusätzlich zum räumlichen Teil der Wellenfunktion. In Bra-Ket Notation angeschrieben folgt für den differentiellen Wirkungsquerschnitt allgemein:

$$\frac{d\sigma}{d\Omega} = \frac{k_s}{k_i}\left(\frac{m_n}{2\pi\hbar^2}\right)^2 \left|\langle\sigma_s k_s m_s|\hat{V}|\sigma_i k_i m_i\rangle\right|^2 \quad (3.48)$$

$$\hat{V} = -\vec{\mu}_n \vec{H}_e \quad (3.49)$$

Für das Vektorpotential eines Dipolfeldes folgt aus dem klassischen Elektromagnetismus:

$$A_e = \frac{\mu_0}{4\pi}\frac{\vec{\mu}_e \times \vec{r}}{|\vec{r}|} = \frac{\mu_0}{4\pi}\vec{\mu}_e \times \vec{\nabla}\frac{1}{|\vec{r}|} \quad (3.50)$$

$$\vec{H}_e = \vec{\nabla} \times \vec{A}_e = \frac{\mu_0}{4\pi}\vec{\nabla} \times \left(\vec{\mu}_e \times \vec{\nabla}\frac{1}{|\vec{r}|}\right) \quad (3.51)$$

Und damit weiters für den Wirkungsquerschnitt:

$$\frac{d\sigma}{d\Omega} = \frac{k_s}{k_i}\left(\frac{m_n}{2\pi\hbar^2}\right)^2 (2\gamma\mu_N\mu_B)^2 \left|\left\langle\sigma_s k_s m_s\left|\vec{\sigma}\cdot\vec{\nabla}\times\left(\vec{s}_e \times \vec{\nabla}\frac{1}{|\vec{r}|}\right)\right|\sigma_i k_i m_i\right\rangle\right|^2 \quad (3.52)$$

Durch Einführen einer Hilfsvariable $\vec{p}$ für die Integration

$$\frac{1}{(2\pi)^3}\int d\vec{r} e^{i(\vec{p}-\vec{Q})\vec{r}} = \delta(\vec{p}-\vec{Q}) \quad (3.53)$$

folgt für das Element:

$$\begin{aligned}&\left\langle k_s\left|\vec{\nabla}\times\left(\vec{s}_e \times \vec{\nabla}\frac{1}{|\vec{r}|}\right)\right|k_i\right\rangle \\ &= \frac{1}{(2\pi)^2}\int d\vec{r} e^{-i\vec{Q}\vec{r}}\int d\vec{p}\hat{p}\times(\vec{s}_e - \hat{p})e^{i\vec{p}\vec{r}} \\ &= 4\pi\hat{Q}\times(\vec{s}_e\times\hat{Q}) \equiv 4\pi\vec{s}_{e\perp}\end{aligned} \quad (3.54)$$

Mit $\vec{s}_{e\perp}$ wird hier die Projektion des Elektronenspins normal zum Vektor $\vec{Q}$ bezeichnet. Durch Zusammenfassen der Vorfaktoren und Ersetzen von $\mu_0 = \frac{1}{\varepsilon_0 c^2}$ sowie dem klassischen Elektronenradius $r_0 = \frac{e^2}{m_e c^2}$, ergibt sich schließlich für den differentiellen Wirkungsquerschnitt:

$$\frac{d\sigma}{d\Omega} = (\gamma r_0)^2 |\langle \sigma_s m_s | \vec{\sigma} \cdot \vec{s}_{e\perp} | \sigma_i m_i \rangle|^2 = (\gamma r_0)^2 \left[1 - \left(\hat{\eta} \cdot \hat{Q}\right)^2\right] \qquad (3.55)$$

Der letzte Schritt in Gleichung (3.55) ergibt sich durch Einsetzten von $\hat{\eta}$, dass die Richtung des magnetischen Moments in der Probe bezeichnet, die Gleichung selbst basiert auf einem Modell mit einem einzelnen Elektron, bei Verallgemeinerung auf ein Atom ergibt sich für den differentiellen Wirkungsquerschnitt:

$$\frac{d\sigma}{d\Omega} = (\gamma r_0)^2 |\hat{\eta}_\perp|^2 \left|f(\vec{Q})\right|^2 = (\gamma r_0)^2 \left[1 - \left(\hat{\eta} \cdot \hat{Q}\right)^2\right] \left|f(\vec{Q})\right|^2 \qquad (3.56)$$

mit dem magnetischen Formfaktor

$$f(\vec{Q}) = \frac{1}{2\mu_B} \int d\vec{r} M(\vec{r})\, e^{-i\vec{Q}\vec{r}}, \qquad (3.57)$$

wobei mit $\vec{M}(\vec{r}) = M(\vec{r})\hat{\eta}$ die Magnetisierungsdichte, die durch den Spin und die Bahnbewegung aller ungepaarten Elektronen im Atom entsteht, definiert ist.

3.2.5 Einflüsse realer Systeme

3.2.5.1 Allgemeines

Die vorangegangenen theoretischen Überlegungen sind unter der Annahme sehr vereinfachter Systeme sowie für ideale Bedingungen zutreffend, in realen Anwendungen sind diese Grundlagen durch weitere Effekte überlagert. Dazu zählen:

- Interpartikulare Interferenzeffekte
- Mehrfachstreuung beim Probendurchtritt
- Polydispersivität
- Effekte durch poröse oder fraktal aufgebaute Materialien

In den meisten Fällen können diese Effekte durch mathematische Modellierung sehr gut beschrieben werden, im Folgenden soll aber nur der Einfluss der Polydispersivität beschrieben werden.

3.2.5.2 Polydispersivität

In realen Anwendungen ist das Streumedium durch Partikel aufgebaut, die unabhängig von ihrer Form eine gewisse Größenverteilung aufweisen, mathematisch gesehen entspricht das einer Verteilung um einen Mittelwert. Da die Wirkungsquerschnitte funktional vom Radius der Streuer abhängig sind, wirkt sich eine Größenverteilung natürlich auf die Streuergebnisse aus. Der Einfluss dieser Verteilung auf eine charakteristische Streufunktion wird als „Verschmieren“ bezeichnet, da sie die Extrema der Funktion aufweicht. Unter der Annahme einer auf 1 normierten Verteilungsfunktion $f(R)$ für die Größenverteilung

$$\int_0^\infty dR\, f(R) = 1 \qquad (3.58)$$

kann der differenzielle Wirkungsquerschnitt größenverteilter Partikel angeschrieben werden mit

$$\frac{d\sigma(Q)}{d\Omega} = (\Delta\rho)^2 \int_0^\infty dR\, f(R)\, V^2(R) |F(QR)|^2 \qquad (3.59)$$

In der Literatur werden verschiedene mathematische Funktionen als geeignete Verteilungsfunktionen beschrieben, dazu zählen die Gaußverteilung, die Log-Normal Verteilung und die Schulz-Verteilungsfunktion [9]. Im Rahmen dieser Diplomarbeit wurde nur die Gaußverteilung verwendet, die wie folgt definiert ist:

$$f(R) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(R-\bar{R})^2}{2\sigma^2}} \qquad (3.60)$$

In Gleichung (3.60) wird der durchschnittliche Teilchenradius mit $\bar{R}$ und seine Standardabweichung mit σ bezeichnet.

Zur Veranschaulichung sind in Abbildung 3.9 simulierte Streufunktionen mit Gaußschen Größenverteilungsfunktionen dargestellt.

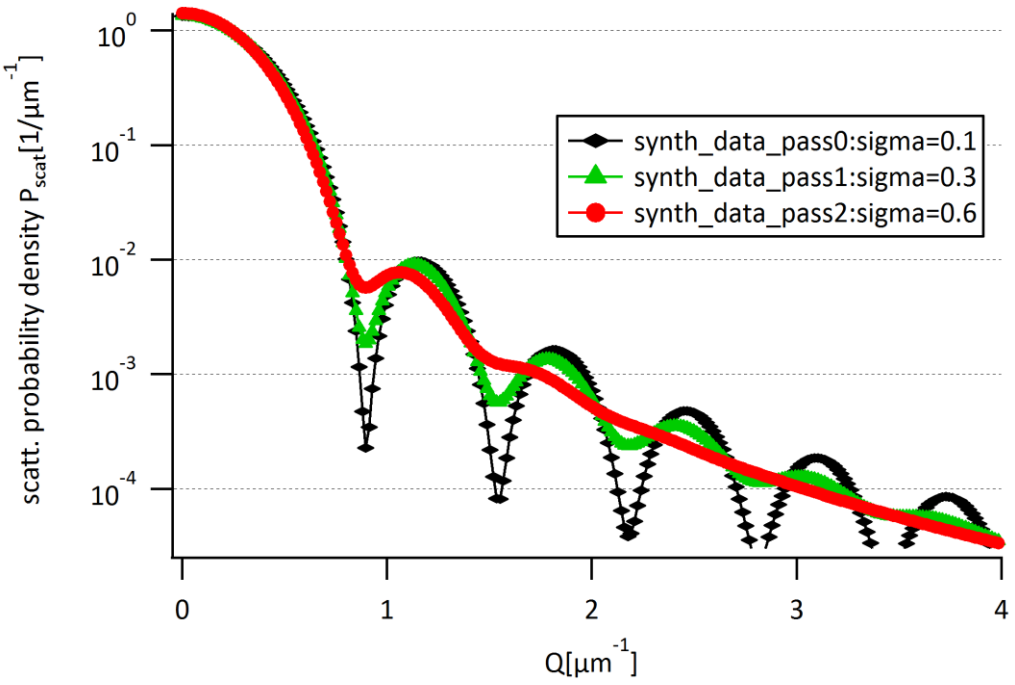

Abbildung 3.9: Einfluss der Breite der Größenverteilung auf die Streufunktion von kugelförmigen Streuern mit einem durchschnittlichen Radius von R =5 µm

Deutlich zu erkennen ist das verstärkte „Verschmieren“ der charakteristischen Streukurve mit steigender Standardabweichung vom Durchschnittsradius. Ebenso zu erkennen ist eine Verschiebung der Scheitelpunkte der lokalen Minima und Maxima hin zu kleineren Q-Werten.

3.2.6 Instrumentenabhängige Einflüsse

3.2.6.1 Schlitzverschmierung

Bei der Durchführung von USANS-Messungen unter Verwendung einer Doppelkristallanordnung tritt, ähnlich wie bei der Polydispersivität, ein weiterer Verschmierungseffekt, die so genannte „Schlitzverschmierung“ der Streukurven auf. Der Begriff hat seinen Ursprung in den historischen Anfängen der Röntgen-Kleinwinkelstreuung, als Streubilder noch mit Hilfe eines verfahrbaren Zählrohrs aufgenommen wurden, vor dem vertikale oder horizontale Schlitzblenden als Kollimationssystem angebracht waren. Die Messdaten dieser Bauform, werden wesentlich von der Geometrie des Schlitzes, also Schlitzhöhe und Schlitzbreite, beeinflusst. Durch die Entwicklung von zweidimensionalen positionsempfindlichen Detektoren (PSD) wurde diese Architektur allerdings abgelöst. Trotzdem ist der Verschmierungseffekt bei USANS-Messung nach wie vor zu beobachten, der Grund dafür liegt in der Reflexionsakzeptanz der verwendeten Siliziumnutenkristalle [17], die in vertikaler Richtung um etwa drei Größenordnungen größer ist als in horizontaler Richtung. Das entspricht im Wesentlichen einer Messung mit einer sehr schmalen und überproportional hohen Schlitzblende. Bei USANS Messungen wird die „klassische“ Schlitzbreite durch die Breite

der Leerkurve des Doppelkristallspektrometers bestimmt, die Schlitzhöhe durch die Winkelakzeptanz in vertikaler Richtung. Der differenzielle Wirkungsquerschnitt für eine schlitzhöhenverschmierte Anordnung hat folgende mathematische Form:

$$\left(\frac{d\sigma(Q_y)}{d\Omega}\right)_{SH} = \int_{-\infty}^{\infty} dQ_z \left(\frac{d\sigma\left(\sqrt{Q_y^2 + Q_z^2}\right)}{d\Omega}\right) \qquad (3.61)$$

Das Koordinatensystem wurde so angelegt, dass der Neutronenstrahl in Richtung der x-Achse einfällt, das Signal dann über die z-Achse integriert wird, und die aufgezeichnete Intensität eine Funktion entlang der y-Ache darstellt.

3.2.6.2 Schlitzbreitenverschmierung

Der Einfluss der instrumentenabhängigen Leerkurve, der als „Schlitzbreitenverschmierung“ bezeichnet wird, entspricht mathematisch einer „Faltung“ des unverschmierten Signals mit der Leerkurve. Die Faltung zweier Funktionen $f, g: \mathbb{R}^n \to \mathbb{C}$ ist allgemein gegeben durch:

$$(f * g)(x) = \int_{\mathbb{R}^n} f(\tau) g(x - \tau)\, d\tau \qquad (3.62)$$

Da die Signalintensität proportional zum differenziellen Wirkungsquerschnitt $I(Q) \sim \frac{d\sigma(Q)}{d\Omega}$ ist, kann für eine schlitzbreitenverschmierte Intensität I_{SB} geschrieben werden:

$$I_{SB}(Q) = \int_{-\infty}^{\infty} I_0(Q_y)\, I(Q - Q_y)\, dQ_y \qquad (3.63)$$

Hier bezeichnet $I_0(Q_y)$ die Auflösungsfunktion und $I(Q)$ die ideale Modellstreufunktion. Durch die Schlitzbreitenverschmierung kommt es bei einer idealen Streufunktion zu einem teilweisen „Auswaschen“ der Maxima und Minima, einem Effekt der auch bei der zuvor behandelten Polydispersivität beobachtet wird.

4 Streudatensynthese mit IGOR Pro

4.1 Diskussion der verwendeten Software – IGOR Pro

Für die Durchführung der Datensynthese, die ein Kernbestandteil dieser Diplomarbeit ist, musste eine Plattform gefunden werden, die Unterstützung für die numerischen Berechnungen und deren Darstellung in graphischer Form bietet. Dafür wurde auf die Software IGOR Pro zurückgegriffen, die bereits in der Neutronengruppe des Atominstituts für wissenschaftliche Zwecke eingesetzt wird. Im Folgenden werden die Software und ihre Funktionsweise kurz beschrieben:

IGOR Pro wird von der Firma Wavemetrics (Weblink: www.wavemetrics.com) entwickelt und ist für die Betriebssysteme Windows und Macintosh verfügbar. Die Einsatzgebiete der Software umfassen wissenschaftliche Datenanalyse, Signalverarbeitung, Kurvenapproximation (Curve Fitting) und digitale Bildbearbeitung sowie -analyse. Mit verhältnismäßig geringem Aufwand können publikationsreife Graphiken erstellt werden, was speziell im wissenschaftlichen Bereich von Interesse ist. Des Weiteren bietet IGOR Pro eine eigene, skriptartige Programmiersprache die, kombiniert mit einem eigens darauf angepassten Compiler, viele, und wenn nötig auch sehr komplexe Anwendungsgebiete abdecken kann. Im Fall dieser Diplomarbeit wurden hauptsächlich die Programmierbarkeit zur Durchführung numerischer, computergestützter Berechnungen und die gebotenen Möglichkeiten zur graphischen Darstellung verwendet.

Es wurde IGOR Pro in Version 6.22A verwendet; eine Lizenz wurde vom Atominstitut bereitgestellt.

Die Einarbeitung in die spezifische Programmiersprache erfolgte mit Hilfe der internen Dokumente, die nach der Installation von IGOR Pro verfügbar sind.

4.1.1 Schlüsselkonzepte

IGOR Pro Anwendungen setzen sich aus folgenden grundlegenden Objekten zusammen [18]:

- Waves
- Graphen

- Tables
- Page Layouts

Da Waves eine zentrale Rolle in dieser Arbeit spielen, soll kurz auf dieses Konzept eingegangen werden:

Das Wave Objekt ist der universell einsetzbare Datenbehälter in IGOR Pro. „Wave“ steht dabei als Kurzform für den Begriff Wellenform, da IGOR Pro in seiner ursprünglichen Form dafür konzipiert wurde, mit Wellenformdaten (z.B. von einem digitalen Oszilloskop, Digital-Analog-Konverterkarten oder anderen wissenschaftlichen Instrumenten) zu arbeiten.

Eine Wave ist vergleichbar mit einem Table in einer Datenbank, er besitzt jedoch eine spezielle auf Wellenformen ausgerichtete Struktur. Jede „Row“ der Wave entspricht einem Datenpunkt und beinhaltet ein- bis mehrdimensionale Datenwerte zu diesem Punkt. Waves können mit einer frei wählbaren Anzahl an Datenpunkten erstellt werden. Die „Columns“-Struktur einer Wave besteht in der Reihenfolge aus: einer Spalte der indizierten Datenpunkte, dem sogenannten „x-Scaling“ – den berechneten, inkrementellen Werten für die x-Achse und anschließend, je nach Anzahl der Dimensionen aus den n-fachen Datenwerten.

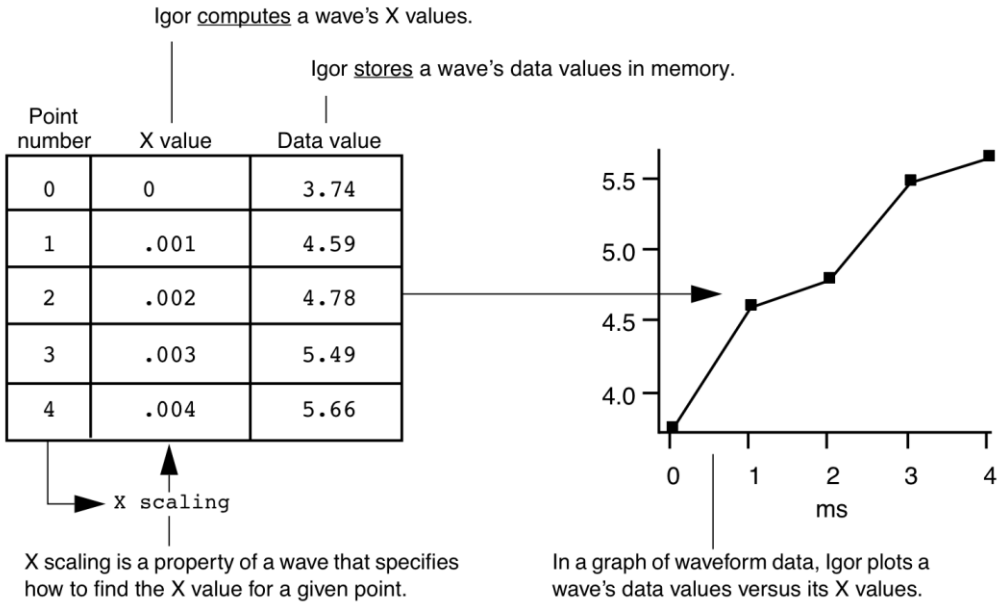

Abbildung 4.1: Illustration des zentralen Wave-Konzepts in IGOR Pro, Auszug aus der internen Dokumentation [18]

Abbildung 4.1 zeigt eine Wave mit 5 Datenpunkten, indiziert von 0 bis 4. Die Skalierung der X-Achse wurde so vorgenommen, dass die Werte bei 0 beginnen und zum vorigen Datenpunkt jeweils ein Inkrement von 0.001 aufweisen. Der Graph auf der rechten Seite zeigt eine Darstellung

der gespeicherten Datenwerte der Wave, aufgetragen gegen die berechneten x-Werte. In diesem Beispiel wird die x-Achse als Zeitachse verwendet, die Werte auf dieser Achse wurden in [ms] aufgetragen.

Das „Scaling“ der Wave erfolgt über die Eingabe eines Anfangspunktes A und eines Endpunktes B. Mit der, ebenfalls vom User festgelegten Anzahl der Datenpunkte N ergibt sich dann das Inkrement auf der X-Achse zu:

$$\Delta X = \frac{(B - A)}{N} \qquad (4.1)$$

IGOR Pro hat darüber hinaus auch die Fähigkeit mit Waves zu arbeiten die nicht diesem Schema mit einheitlichem Abstand auf der x-Achse entsprechen. Dabei behandelt das Programm zwei einzelne Waves, die jeweils eine Dimension beschreiben als sogenanntes XY-Paar. Dieser Ansatz ermöglicht die Vorgabe von Werten für beide Achsenabschnitte, eine Anforderung, die in wissenschaftlichen Applikationen häufig anzutreffen ist.

Das Wave-Datenmodell kann in der Ausprägung „Multidimensional Wave“ bis zu vier Dimensionen an Daten speichern. Nicht nur numerische Werte in verschiedenen Genauigkeiten wie „Single“ oder „Double Precision“, sondern auch komplexe Zahlen oder reiner Text kann in IGOR Pro waves gespeichert werden.

4.1.2 Programmierbeispiel mit IGOR Pro Waves

Anhand des folgenden Beispiels soll kurz erklärt werden, wie eine mathematische Funktion mit IGOR Pro synthetisiert werden kann:

Zum Erstellen einer Wave wird die MAKE Operation verwendet; im unten angeführten Beispiel wird eine Wave mit Namen „Testwave“ erstellt, die 100 Datenpunkte enthält. Die Option /D bewirkt, dass zu speichernde Werte im Gleitkommaformat mit doppelter Genauigkeit, der sogenannten „Double Precision“ abgelegt werden. Die Anweisung lautet:

```
MAKE /N=100 /D Testwave
```

Des Weiteren soll die „Testwave“ mit einer Skalierung von 0-5 auf der x-Achse versehen werden, die Einheit kann hier als String mitgegeben werden – in diesem Beispiel „kg“.

```
SETSCALE x 0,5,“kg“, Testwave
```

Die erstellte „Testwave“ enthält noch keine Werte für die y-Achse. Wenn die Werte der y-Achse den Funktionswerten *f(x)* entsprechen, dann kann man alle 100 Datenpunkte über einfaches „Wave Assignment“ synthetisieren lassen. Besteht z.B. eine quadratische Abhängigkeit, wäre der entsprechende Befehl dafür

Testwave=x^2

Dies ist eine verkürzte Schreibweise um das „Wave Assignment“ effizienter zu gestalten. Um den Datenpunkten Werte zuzuordnen kann auf Waves alternativ auch mittels

Testwave[p] Datenpunkt einer Wave

Testwave(x) Datenpunkt an der Stelle x

zugegriffen werden.

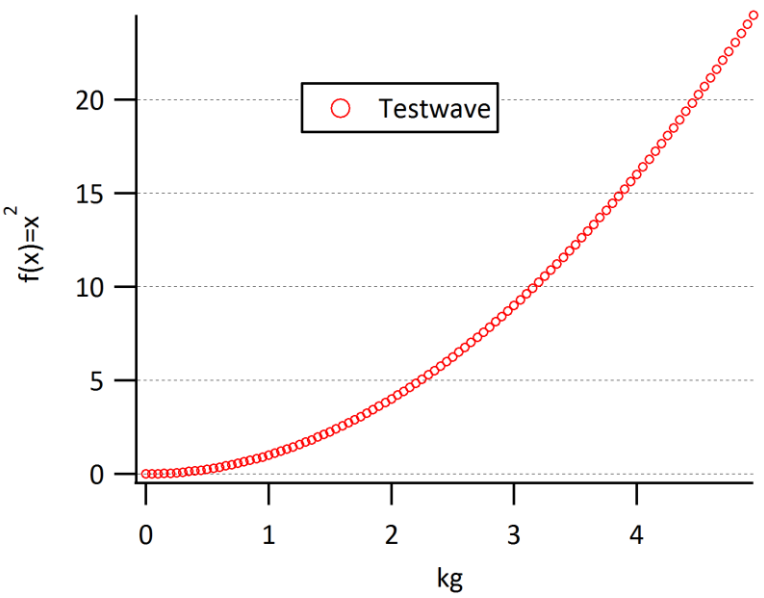

Abbildung 4.2: Graphische Darstellung der Testwave $f(x) = x^2$ aus dem Programmierbeispiel

4.2 USANS Simulation Application

Ziel der USANS Simulation Application ist es, dem User auf Basis von IGOR Pro eine einfache, übersichtliche Oberfläche bereitzustellen, die auf Knopfdruck verschiedene Ablaufstufen der Neutronenstreuung berechnet und anschließend graphisch ausgibt. Da die Berechnungen mitunter sehr zeitintensiv sein können, wird speziell auf die Unterstützung von modernen Computersystemen mit mehrkernigen Prozessoren Rücksicht genommen.

4.2.1 Allgemeines

4.2.1.1 Mehrdimensionale Integration mit der Funktion INTEGRATE1D

Die Funktion INTEGRATE1D bietet die Möglichkeit vordefinierte Funktionen über ein angegebenes Intervall zu integrieren. Ihre Syntaxdefinition lautet [18]:

INTEGRATE1D(UserfunctionName, min_x, max_x, options)

min_x und max_x entspricht den Integrationsgrenzen innerhalb derer die Funktion integriert werden soll, „Options“ bietet die Möglichkeit verschiedene Integrationsalgorithmen zu wählen (Trapezoidal, Romberg, Gaussian Quadrature). Die Funktion kann ebenfalls dafür verwendet werden, um höher-dimensionale Integrale zu berechnen. Am Beispiel der Integration einer zweidimensionalen Funktion *f(x,y)*

$$h = \int_{ymin}^{ymax} \int_{xmin}^{xmax} (3x + 2y + xy)\, dx\, dy \qquad (4.2)$$

wird ersichtlich, dass für die Berechnung zwei Userfunktionen verschachtelt werden müssen:

```
Function do2dIntegration(xmin,xmax,ymin,ymax)
    Variable xmin,xmax,ymin,ymax

    Variable/G    globalXmin=xmin
    Variable/G globalXmax=xmax
    Variable/G globalY

    return  Integrate1d(userFunction2,ymin,ymax,1)   // Romberg integration
End

Function userFunction1(inX)
    Variable inX

    NVAR globalY=globalY

    return (3*inX+2*globalY+inX*globalY)
End

Function userFunction2(inY)
    Variable inY

    NVAR globalY=globalY
    globalY=inY
    NVAR globalXmin=globalXmin
    NVAR globalXmax=globalXmax

    return Integrate1D(userFunction1,globalXmin,globalXmax,1)          // Romberg integration
End
```

Abbildung 4.3: IGOR Codeausschnitt für die Berechnung der mehrdimensionalen Integration aus Gleichung (4.2)

4.2.1.2 Multithreading in IGOR

Alle modernen Computer- und Betriebssysteme bieten Unterstützung für Multicore- oder Multithreadingarchitekturen an, deren großer Vorteil in der parallelen Verarbeitung von einzelnen Prozessen liegt. Schon längst bestimmt nicht mehr das Rennen um die höchste Taktfrequenz die Leistungsfähigkeit von Prozessoren, sondern die Kombination aus Takt und Anzahl der Kerne. Ihre Anzahl, und damit die Fähigkeit, Daten parallel zu verarbeiten, steigt in den letzten Jahren stetig, wodurch es einen großen Spielraum für Performanceverbesserungen in diesem Bereich gibt. IGOR Pro bietet verschiedene Mechanismen an, um die Durchführung von Programmteilen auf Multicore-Systemen zu beschleunigen. Die einfachste Variante, die auch beim USANS Simulator zum Einsatz kommt, ist das Multithreading-gestützte Wave Assignment. Das Einfügen des Keywords „Multithread“ vor einem Wave Assignment aktiviert die Funktionalität im Compiler:

MULTITHREAD Wave=expression

Dem Vorteil der beschleunigten Abarbeitung stehen Nachteile in Form von verschiedenen Restriktionen für die „Expression“ gegenüber. Alle Funktionen, die beim Wave Assignment aufgerufen werden, müssen „Threadsafe“ sein [18]. Werden in diesem Falle User-Funktionen verwendet, so müssen diese mit dem Keyword THREADSAFE gekennzeichnet werden:

```
ThreadSafe Function myadd(a,b)
    Variable a,b

    return a+b
End
```

Abbildung 4.4: IGOR Codeausschnitt für die Definition einer Threadsafe-Funktion. Abweichungen von diesem Schema führen zu Compiler-Fehlern

Funktionen, die diesem Schema entsprechen, können bei simultaner Verarbeitung in mehreren Threads korrekt ablaufen. Im Taskmanager des Betriebssystems ist bei Verwendung dieser Routinen deutlich die Verwendung aller Prozessorkerne zu erkennen. Erste Tests mit isolierten Wave Assignments ergaben einen Performancegewinn um den Faktor x2 bis x3, weiter nach oben skalierend in Abhängigkeit von der Anzahl der Punkte einer Wave und der Komplexität der Anweisungen die in der „Expression“ enthalten sind.

4.2.1.3 Performancemonitoring und Logging

Da es bei der Durchführung gewisser Funktionen erwartungsgemäß zu sehr langen Berechnungszeiten kommen kann, soll in der Applikation eine Möglichkeit zur Beurteilung der Performance geschaffen werden.

Alle gekapselten Funktionen der USANS Simulation Application wurden mit Time-Monitoring Routinen versehen, um ein Gefühl für die Skalierung der Berechnungszeit in Bezug auf die Input-Parameter zu bekommen. Der Beispielcode für eine solche Routine mit Returntext formatiert in Sekunden sieht wie folgt aus:

```
variable timerrefnum
variable etime
timerrefnum=startMStimer

[timeconsuming expression]

etime=stopMStimer(timerrefnum)
print "elapsed time:"+num2str(etime/1e6)+"s"
```

Abbildung 4.5: IGOR Codeausschnitt einer Funktion, mit der die Durchführungszeit eines Statements erfasst wird, hier beispielhaft „timeconsuming expression“ genannt, Ausgabe formatiert in [s]

Besonders zeitintensive Berechnungen, die z.B. ab der Synthese der kombinierten Polydispersivität und Schlitzhöhenverschmierung zu erwarten sind, wurden mit einer Verlaufsanzeige ausgestattet, um den User über den Status des Berechnungsfortschritts zu informieren. Ein weiteres Feature betrifft die Übersicht bereits erfolgter Durchläufe: Jede Funktion schreibt Informationen betreffend seiner Durchführung inklusive der wichtigsten Parameter in den History-Bereich der Applikation (siehe Abbildung 4.6). Dieses als Logging bezeichnete Verfahren gibt dem User die Möglichkeit, Fehler im Ablauf aufzuspüren und bietet zudem eine praktische Verlaufsansicht zur Orientierung.

```
  ---
Initialize
  ---
USANS Synthesis Simple Variant generation time(P_up):114.21s
  USANS Synthesis Simple Variant generation time(P_down):114.53s
  Generated RockingCurve: -4/4;250 points
  Norming to Sourcewave: _free_, Factor=0.0021786
  Norming to Sourcewave: _free_, Factor=5.8203e-07
  Convolving _free_ with generated rockingcurve
  Generated RockingCurve: -4/4;250 points
  Norming to Sourcewave: _free_, Factor=0.0058613
  Norming to Sourcewave: _free_, Factor=5.8202e-07
  Convolving _free_ with generated rockingcurve
  Generated RockingCurve: -4/4;250 points
  Summing Scattering Curve _free_ with generated RockingCurve, wstreu=3%
  Norming to Sourcewave: _free_, Factor=0.99348
  Generated RockingCurve: -4/4;250 points
  Summing Scattering Curve _free_ with generated RockingCurve, wstreu=3%
  Norming to Sourcewave: _free_, Factor=0.99348
  Norming area to value:1, Wavename: synth_data_pass0, Factor=1.0931e+05
  Showing Wave Graph: synth_data_pass0
•SetAxis/A
•SetAxis/A
•SetAxis/A
  ---
Initialize
  ---
Form Factor Spherical Model generation time: 0.00049496s
  Norming area to value:1, Wavename: synth_data_pass0, Factor=2.0669e+05
  Showing Wave Graph: synth_data_pass0
```

Abbildung 4.6: Auszug aus dem History-Bereich der USANS Simulation Application. Beispielhaft wurde nach dem Initialisieren der Applikation eine USANSPOL Streukurve (Simple Variant) erstellt. Im Anschluss daran wurde eine einfache Streufunktion synthetisiert.

4.2.2 Konventionen zur Datenausgabe

SANS Experimentergebnisse liegen in Form von Streukurven vor, die gemessene Intensitäten (bedingt durch das Funktionsprinzip des Detektors in Counts/s) in Abhängigkeit des Streuvektors *Q* darstellen. Im Prinzip ist diese Streukurve eine Abbildung des, in den vorigen Kapiteln erwähnten, Wirkungsquerschnitts. Da der Kurvenverlauf, der sich meist über mehrere Größenordnungen erstrecken kann, gleichzeitig Informationen über die Probenstruktur beinhaltet, wird die Intensität bzw. der Wirkungsquerschnitt zur verbesserten Darstellung auf einer logarithmischen Skala aufgetragen.

Um die erstellten Graphen untereinander besser vergleichen zu können, werden die Funktionswerte üblicherweise einem Normierungsverfahren zugeführt. In der USANS Simulation Application wird eine einheitliche Darstellung durch die Wahl einer passenden Normierungskonstanten, die im Weiteren mit C_N bezeichnet wird, erreicht. Mathematisch kann dieses Verfahren folgendermaßen angeschrieben werden:

$$P_{scat}(Q) = \frac{1}{C_N} I(Q) \qquad C_N = \int I(Q)dQ$$

Die Funktionswerte können nach Normierung folgendermaßen interpretiert werden: Da die Fläche unter der Kurve mit der Anzahl der Neutronen korreliert, und diese nun auf 1 normiert wurde, entspricht jeder Funktionswert der Wahrscheinlichkeitsdichte ein (einzelnes) Neutron nach der Streuung an diesem Q-Wert vorzufinden.

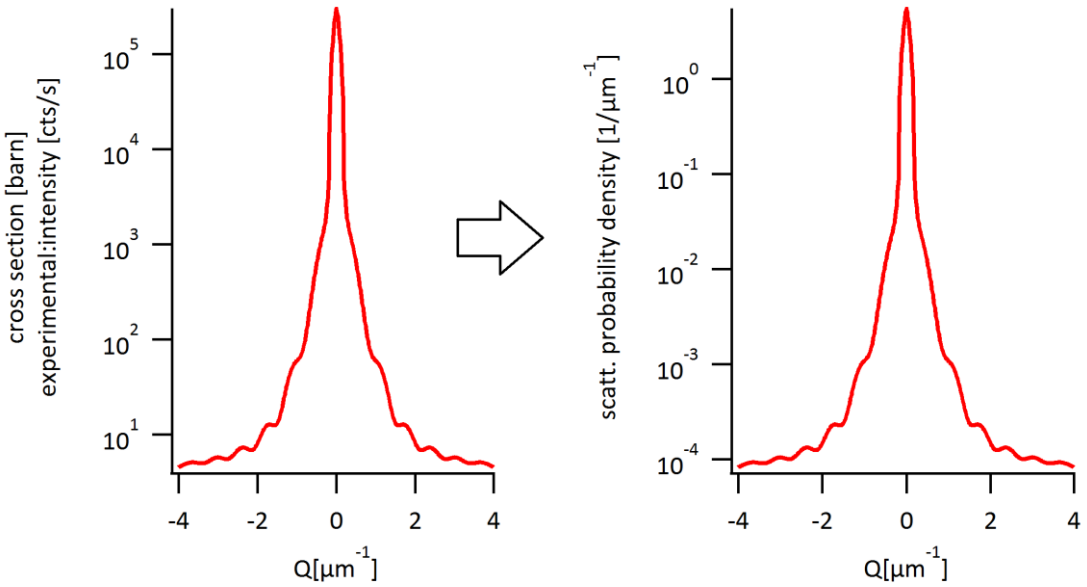

Abbildung 4.7: Datenausgabe - Übergang zur Darstellung der Wahrscheinlichkeitsdichte durch Normierung der Funktionswerte anhand einer Beispielfunktion

4.2.3 Kopfteil – Generelle Parameter und Reset-Funktion

Am Kopfteil der Applikationsoberfläche befindet sich ein Panel, dass, ausgestattet mit Eingabefeldern mehrere allgemeine Parameter steuert, die für alle Simulationsschritte gleichermaßen relevant sind. Der Button „Reset Application“ löscht alle vorhandenen Simulationen/Waves aus dem Speicher und setzt jeden einzelnen Parameter auf seinen Standardwert zurück. Weiter findet man im dem als „General Synthesis Parameters“ benannten Panel Einstellungsmöglichkeiten für: Qmax und Qmin legen die Ober- und Untergrenze des gewünschten Beobachtungssintervalls auf der Q-Achse fest, der Parameter „Range Resolution“ bestimmt die Anzahl der Datenpunkte im gewählten Intervall und ist ein Maß für die graphische Auflösung der erzeugten Streukurve, die andererseits auch höhere Berechnungszeiten erfordert. Die Höhe des Zahlenwerts im Eingabefeld „Intensity Equivalent“ steuert die Stärke des statistischen Rauschens, das durch Aktivieren der Checkbox „Add gaussian noise“ einem Simulationsdurchlauf hinzugefügt werden kann.

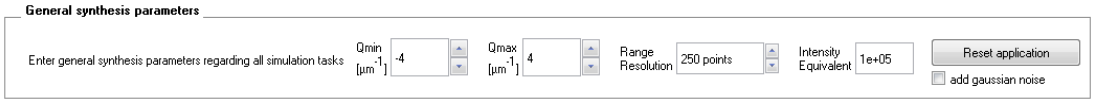

Abbildung 4.8: Kopfteil der Oberfläche mit allgemeinen Parametern und Funktionen

4.2.4 Simulation der Streufunktion, ideale Streukurve

Die grundlegendste Funktionalität des Simulators, die später auch in allen weiteren Prozeduren verwendet wird, ist die Synthetisierung einer idealen Streukurve bzw. der reinen Streufunktion. Die theoretische Grundlage zur Berechnung des Formfaktors wurde bereits in Abschnitt 3.2.1 vorgestellt. Im Rahmen dieser Diplomarbeit wurden ausschließlich Formfaktoren mit sphärischen Streufunktionen (siehe Kapitel 3.2.2.1), die nur von einem Parameter, nämlich dem Radius des Streuobjekts, abhängig sind, betrachtet. Für die Datensynthese wird folgende Funktion verwendet:

$$P_{scat}(Q) = \frac{1}{C_N} V_p^2 \left(3 \frac{(\sin QR - QR \cos QR)}{(QR)^3} \right)^2 \qquad (4.3)$$

Wird Gleichung (4.3) numerisch berechnet, so tritt an der Stelle *Q=0* ein Problemfall auf, die Funktion liefert einen ungültigen Wert, da es sich hier um einen Grenzwert der unbestimmten Form "$\frac{0}{0}$" handelt. An dieser Stelle muss der Funktion durch eine Ausnahmeroutine ein gültiger Wert, unabhängig vom Parameter *Q,* zugewiesen werden. Der Funktionswert an dieser Stelle kann durch eine analytische Lösung berechnet werden, in diesem Fall erhält man durch vierfache Anwendung der Regel von L'Hospital den Grenzwert

$$\lim_{Q \to 0} \left(V_p^2 \left(3 \frac{(\sin QR - QR \cos QR)}{(QR)^3} \right)^2 \right) = V_p^2 \qquad (4.4)$$

Nachdem das Ergebnis (4.4) bei der Funktionsdefinition in IGOR berücksichtigt wurde, ergab sich ein lückenloser Werteverlauf und weiterer Folge eine stetige Streukurve bei *Q=0*. Die entsprechenden Bedienungselemente und ein Eingabefeld für den Partikelradius befinden sich auf dem Übersichtspanel im Bereich „Data synthesis - Scattering Function".

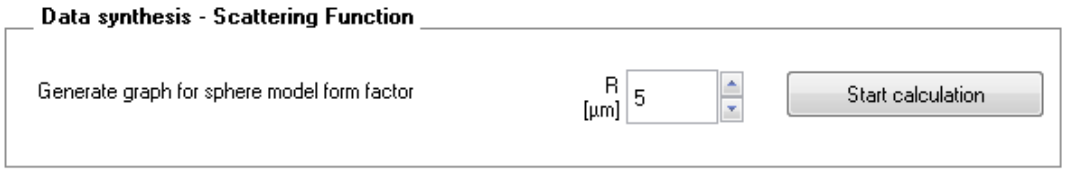

Abbildung 4.9: Oberflächenausschnitt der USANS Simulation Application für die Synthese der einfachen Streufunktion

Durch Betätigen des Buttons „Start calculation“ werden folgende Prozesse ausgelöst:

- Erstellen eines Wave-Objekts im Intervall [Q_{min},Q_{max}]
- Synthetisieren der Kurvendaten durch Wave Assignment
- Ausgabe in einem Graph-Objekt

Bei Verwendung des Referenzsystems liegt die Zeit zum Berechnen der synthetischen Daten selbst bei sehr großen Punktemengen im Bereich von wenigen Millisekunden. Abbildung 4.10 zeigt das Ergebnis der Funktion anhand seiner graphischen Darstellung.

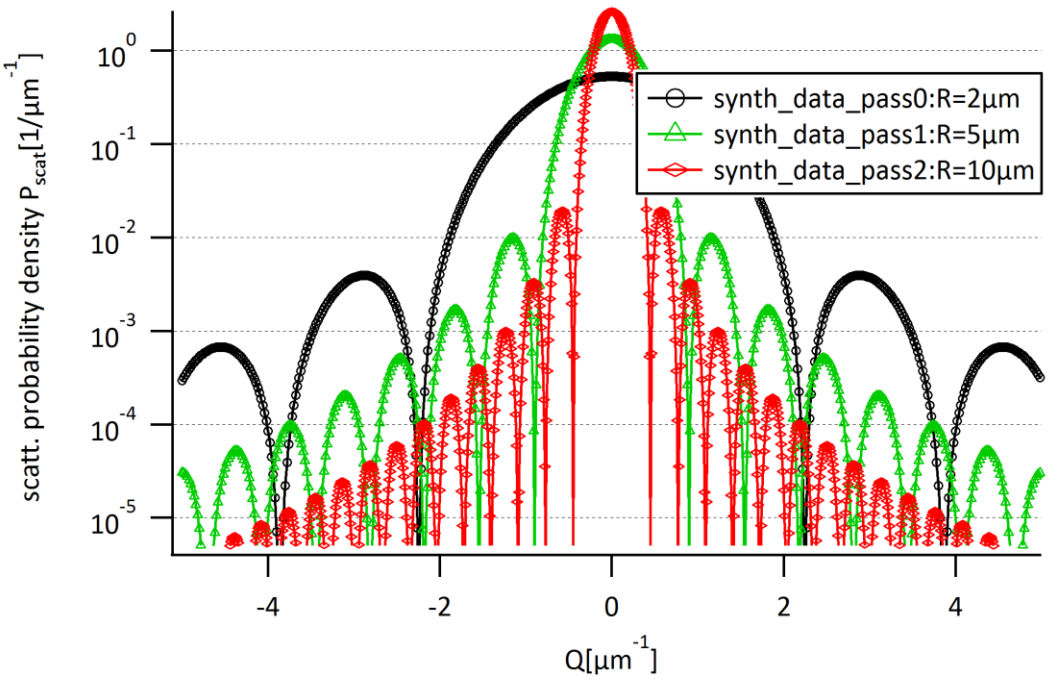

Abbildung 4.10: Synthetische Streukurven von sphärischen Objekten mit unterschiedlichen Radien

4.2.5 Simulation der Schlitzhöhenverschmierung

Die theoretischen Grundlagen der Schlitzhöhenverschmierung wurden bereits im Kapitel 3.2.6.1 beschrieben. Im Wesentlichen kommt es zu einem „Verschmieren“ der Streukurve, das nicht durch die Verwendung von Schlitzblenden, sondern bedingt durch den experimentellen Aufbau mit Doppelkristallanordnung verursacht wird. Dem mathematischen Äquivalent dieses Effekts

entspricht eine Integration des Streuvektors $\hat{Q}$ über die Schlitzhöhe, definitionsgemäß in Richtung Q_z. Die Synthetisierungsfunktion der verschmierten Streukurve bei sphärischer Streugeometrie ist gegeben durch:

$$P_{scat}(Q) = \frac{1}{C_N} \int_{Qmin}^{Qmax} dQ_z \, V_p^2 \left(3 \frac{\left(\sin \hat{Q}R - \hat{Q}R \cos \hat{Q}R\right)}{\left(\hat{Q}R\right)^3} \right)^2 \qquad (4.5)$$

$$\hat{Q} = \sqrt{Q_y^2 + Q_z^2} \qquad (4.6)$$

Werden numerische Verfahren zur Berechnung von Integralen eingesetzt, wie es hier auch bei IGOR Pro der Fall ist, dann können diese im Allgemeinen nur innerhalb endlicher Integrationsgrenzen gelöst werden. Im Unterschied zur Theorie, die laut Gleichung (3.61) ein Integrationsintervall von -∞ bis ∞ aufweist, müssen die Integrationsgrenzen hier in einem endlichen Bereich zwischen Q_{min} bis Q_{max} angesetzt werden, um eine numerische Lösung zu ermöglichen. In weiterer Folge wird die Obergrenze Q_{max} und die Untergrenze Q_{min} durch geeignete Ansätze so gewählt, dass sich alle relevanten Teile der Kurve, die einen Beitrag zum Integral liefern können, innerhalb des Integrationsintervalls befinden. Da im Rahmen dieser Arbeit ständig neue Erkenntnisse in Bezug auf die internen Abläufe von IGOR Pro gewonnen werden konnten, wurden mehrere Methoden zur Berechnung von schlitzhöhenverschmierten Streukurven entwickelt. Jede beschriebene Variante hat ihren, in Abhängigkeit von den Anforderungen bevorzugten Anwendungsbereich sowie spezifische Vor- und Nachteile.

Um sich den Ausgangspunkt der Integration zu veranschaulichen, wurde mit Hilfe von IGOR Pro ein dreidimensionales Objekt synthetisiert, das die Intensitätsverteilung einer unverschmierten Streufunktion in Richtung Q_y und Q_z (siehe Abschnitt 4.2.4) beinhaltet. Ein 3D-Plot in Abbildung 4.11 zeigt das Ergebnis; seine räumliche Struktur wurde durch eine skalenabhängige Farbverteilung weiter hervorgehoben.

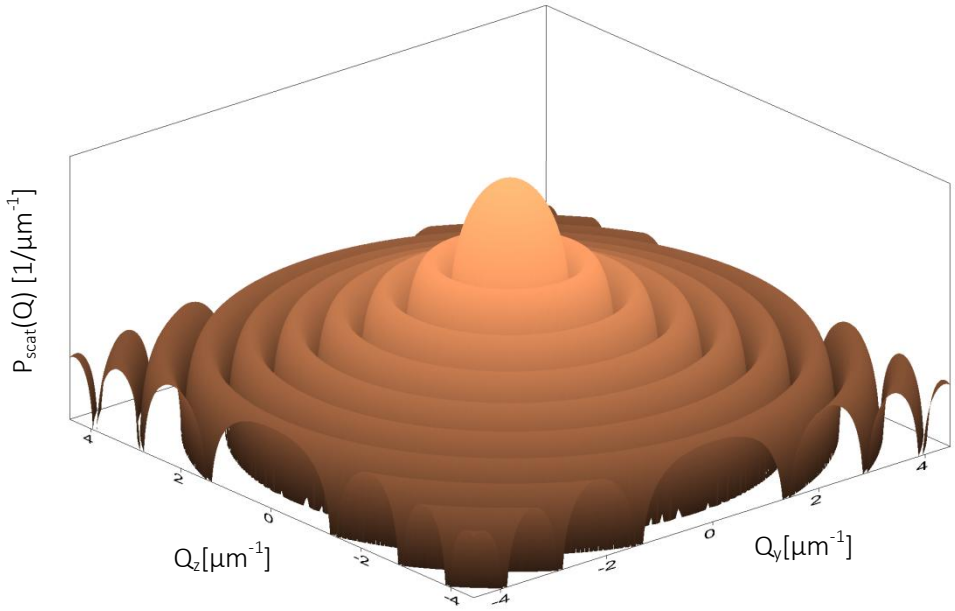

Abbildung 4.11: Farblich hervorgehobene 3D-Darstellung einer unverschmierten Streufunktion mit R = 5 µm. Zur Berechnung der schlitzhöhenverschmierten Intensität muss in jedem Punkt Q_y über Q_z integriert werden.

Variante 1: Eindimensionale Integration mit der integrierten Funktion INTEGRATE1D

Der Vorteil der Funktion INTEGRATE1D ist, dass im Argument neben dem Integrationsintervall direkt eine vordefinierte (analytische) Funktion verwendet werden kann, was prinzipiell dem intuitivsten Ansatz entspricht und sie deshalb für anfängliche Simulationen eingesetzt wurde. Alle anderen Integrationsmethoden in IGOR erwarten die Eingabe des Integranden in Form eines Wave-Objekts, das vor seiner Verwendung natürlich aufgebaut werden muss. Nachteile sind die fehlende Multithreadingunterstützung sowie eingeschränkte Freiheit in Bezug auf Skalierbarkeit, da es sich um eine vorgefertigte Routine handelt. Aufgrund ihrer Einfachheit wurde die Variante speziell zum Debugging, Genauigkeitsvergleich und als Benchmark für die Berechnungsdauer komplexerer Varianten verwendet. Der Programmcode dieser Berechnungsvariante kann dem Abschnitt 7.3 entnommen werden.

Variante 2: Einfacher Integrationsalgorithmus mit Multithreading – „Simple Variant"

Auf dem Weg zur Synthetisierung der kompletten USANSPOL-Streukurve wird es erforderlich sein, eine Kombination aus Schlitzhöhenverschmierung und Polydispersivität zu berechnen. Dieser Vorgang wird, schon vom Prinzip her, ungleich aufwändiger ablaufen, und es wird nötig sein, die Berechnung für jeden Einzelprozess, wie hier die Schlitzhöhenverschmierung, in Hinsicht auf Performance und Skalierbarkeit möglichst zu optimieren. Aufgrund ihrer immanenten Nachteile ist Variante 1 für den Einsatz in solchen Szenarien ungeeignet, die Entwicklung eines eigens dafür entwickelten Integrationsalgorithmus soll dieses Problem lösen und eine Basis für zukünftige Be-

rechnungsschritte schaffen. Abbildung 4.12 stellt den Kernbereich des Algorithmus in einer ersten Fassung dar.

```
//Schleifeninitialisierung
do
	//Zuweisen des x-Werts der wave im Punkt ii
	qy=pnt2x(targetwave,ii)
	//Aufbau Hilfswave
	multithread helperwave=standardsphere(sqrt(qy^2+x^2),r)
	//Integralberechnung und Zuweisung
	targetwave[ii]=area(helperwave)
	//Schleifenzähler
	ii+=1
	//Abbruchbedingung
while(ii<numpnts(targetwave))
```

Abbildung 4.12: Berechnung der Schlitzhöhenverschmierung nach Variante 2; Codeausschnitt einfacher Integrationsalgorithmus mit Multithreading, Ansatz mit DO-WHILE Schleife

Die Funktionsweise basiert auf einer fußgesteuerten DO-WHILE Schleife die alle Punkte (ii) der Ergebnis-Wave „targetwave“ durchläuft bis die WHILE-Bedingung an ihrem Ende erfüllt ist. Im sogenannten „Schleifenrumpf“ werden die einzelnen Punkte der Kurve berechnet: Im ersten Schritt wird die Hilfswave durch Wave Assignment einer Userfunktion, in diesem Fall der sphärischen Streufunktion, aufgebaut. Durch das vorangestellte Keyword wird dieser Vorgang im optimierten Multithread-Modus ausgeführt, was speziell bei Waves mit sehr hoher Punkteanzahl deutliche Vorteile bringt. Im zweiten Schritt wird das Integral der Hilfswave mit der AREA Funktion berechnet und dem entsprechenden Punkt der Ergebniswave zugeordnet.

Nach eingehender Code-Analyse und weiterführendem Studium von [18] konnte der schleifenbasierte Ansatz unter Einbeziehung des Konzepts der multidimensionalen Waves weiter verbessert werden. In seiner finalen Fassung wird zur Berechnung der Funktionswerte eine temporäre, 2-dimensionale Wave(Matrix) aufgebaut, die alle Punkte in einer einzelnen Zuweisung abarbeitet. Das hat den Vorteil, dass jener Teil des Algorithmus, der die meiste Rechenzeit benötigt, noch „dichter“ verarbeitet wird und die Auslastung jedes Prozessorkerns damit fast bei 100% liegt. Die benötigte Integration „faltet“ die Matrix wieder in ein 1-dimensionales Objekt, das schließlich der resultierenden Wave als Wert zugeordnet wird. Abgesehen von der vereinfachten Programmierung liegt der Performancegewinn dieser Variante abhängig von der gewählten Punkteanzahl bei ca. 10-20% im Vergleich zum Schleifenansatz. Nachteil ist der höhere temporäre Speicherbedarf beim Erstellen des Matrixobjektes. Ein Auszug des Kernbereichs des Integrationsalgorithmus in seiner finalen Fassung ist in Abbildung 4.13 dargestellt, der gesamte Code kann Abschnitt 7.3 entnommen werden.

```
//Punkteanzahl festlegen, targetwave=Anzahl der Punkte im Graphen
    variable pts_result=numpnts(targetwave)
//Punkteanzahl festlegen, helperwave=Anzahl der Punkte für die Integration
    variable pts_helperwave_slit=500
//Erstellen eines temporären 2D wave-objekts (Matrix)
    make/N=(pts_result,pts_helperwave_slit)/D/FREE wTemp
//Skalierung der Matrix
    setscale x, (qmin),(qmax),"",wTemp
    setscale y, (10*qmin),(10*qmax),"", wTemp
//Berechnung der Funktionswerte in der Matrix
    multithread wTemp=standardsphere(sqrt(y^2+x^2),r)
//Integration über die y-Ebene(Columns)
    integrate/DIM=1 wTemp
//Ergebnis aus der Matrix der Zielwave zuweisen
    targetwave=wTemp[p][pts_helperwave_slit]
```

Abbildung 4.13: Berechnung der Schlitzhöhenverschmierung nach Variante 2; Codeausschnitt einfacher Integrationsalgorithmus mit Multithreading, Ansatz mit 2-dimensionaler Wave (Matrix)

Bei beiden Ansätzen ist darauf zu achten, dass die benötigten Hilfsberechnungen in ausreichend feiner Granularität, oder, in den Konzepten von IGOR Pro ausgedrückt, mit ausreichend hoher Punkteanzahl durchgeführt werden. Innerhalb der vorgegebenen Parameterintervalle der Applikation wurde die optimale Punkteanzahl in Bezug auf Granularität und Berechnungszeit empirisch ermittelt.

Der hier vorgestellte Integrationsalgorithmus wird deshalb als „einfach“ bezeichnet, weil in seinem Ablauf mit einem konstanten, endlichen Integrationsintervall gearbeitet wird. Optimale Werte für Ober bzw. Untergrenzen wurden empirisch ermittelt und haben sich in Testszenarien bewährt. Details können der Funktion „get_int_limit“ in Abschnitt 7.3 entnommen werden.

Variante 2 nähert sich der Problemstellung, wie bei der Verwendung von IGOR Pro uneigentliche Integrale numerisch abgebildet werden können, vom Gesichtspunkt minimierter Berechnungszeiten her, höhere Genauigkeit und Flexibilität auf Kosten der Berechnungszeit führt zur nachfolgenden Variante 3.

Variante 3: Integrationsalgorithmus mit Multithreading und automatischer Intervallbestimmung – „Complex Variant“

Im hier vorgestellten Algorithmus sollen die deutlichen Vorteile von Multithreading analog zu Variante 2 integriert werden, die empirische Komponente eines fixen Integrationsintervalls pro Hilfswave soll jedoch durch eine dynamische Intervallbestimmung ersetzt werden. Schon vom Ansatz her ist dabei mit einer massiven Erhöhung der Berechnungszeit zu rechnen. Der zu entwickelnde Integrationsalgorithmus soll zusammenfassend folgenden Anforderungen genügen:

- Multithreadingfähig
- Skalierbar
- Dynamische Intervallbestimmung
- Optional: Teilberechnungen einsehbar

Um weitere Erklärungsschritte verständlich zu machen, bedarf es folgender Grundlage: Eine schlitzhöhenverschmierte Streukurve besteht aus einer variablen Anzahl N von berechneten Punkten (siehe Abschnitt 4.2.3). Jeder dieser Punkte ergibt sich aus Integration der Funktion in Gleichung (4.5); Q_z-Werte sind prinzipiell unbeschränkt, Q_y hingegen ist konstant und durch den Zahlenwert des Punktes auf der x-Achse gegeben. In den Terminus von IGOR übersetzt heißt das, dass für die Berechnung jedes Punktes eine eigene Wave aufgebaut werden muss, die nur dem Zwecke der Integration dient. Aus diesem Grund wird im Weiteren von sogenannten „Hilfswaves“ gesprochen. In Abbildung 4.14 werden solche Hilfswaves für verschiedene Punkte derselben Streukurve, die im Bereich (Q[-4/4µm^{-1}]) entwickelt wurde, dargestellt. Ihr Verlauf ist sowohl für positive als auch negative Q-Werte der Streukurve identisch, was aus der Symmetrie der Streukurve um den Wert 0 und mathematisch aus Gleichung (4.6) folgt.

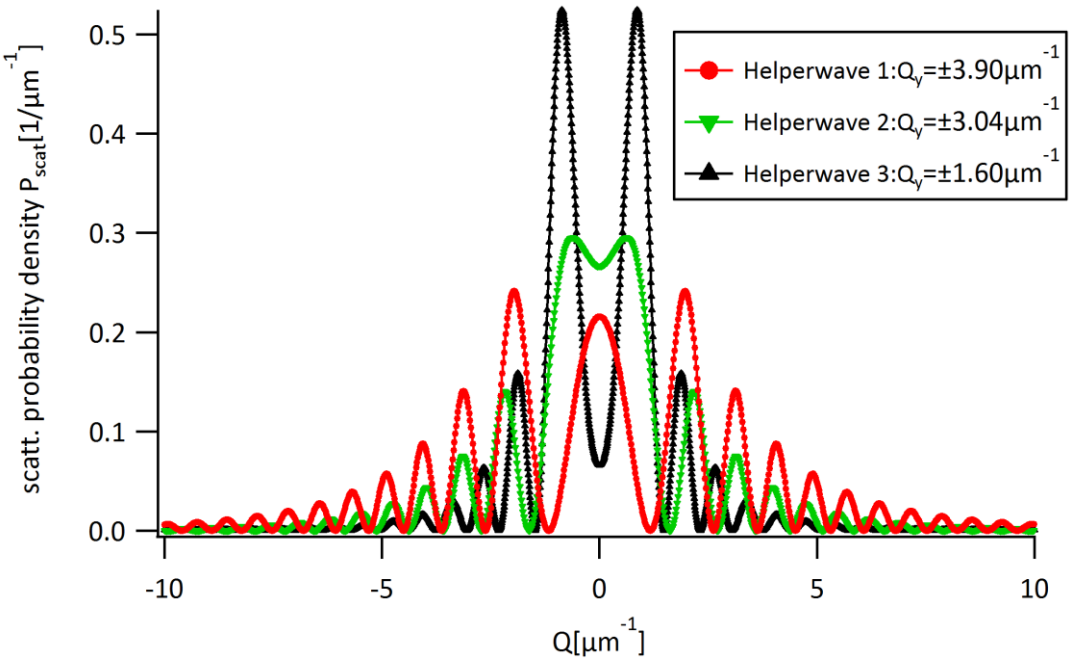

Abbildung 4.14: Darstellung verschiedener normierter Hilfswaves die zum Berechnen der Punkte einer schlitzhöhenverschmierten Streukurve herangezogen werden. Die dargestellten Q-Werte sind den Punkten der x-Achse der Streukurve (Q[-4/4µm^{-1}]) zugehörig.

Es ist deutlich zu erkennen dass Hilfswaves, für die Berechnung von Punkten in außenliegenden Bereichen der Ergebniskurve, wie am Beispiel von Abbildung 4.14 bei Q_y>±3 µm^{-1} zu sehen ist,

einen breiteren Kurvenverlauf mit verhältnismäßig geringerem Maximum aufweisen. Je näher die Hilfswaves für die Berechnung der Punkte um *Q=0* der Ergebniswave herangezogen werden, desto steiler ist der Kurvenverlauf und die Amplituden ihrer Maxima fallen wesentlich höher aus. Setzt man voraus, dass Waves die durch sie abgebildeten Funktionen in ausreichender Granularität auflösen, so kann man diesem Verhalten entnehmen, dass für innenliegende Hilfswaves kürzere Integrationsintervalle nötig sein werden als für außenliegende. Diese Annahme setzt ein periodisches Abklingen der Hilfswave-Funktionen für $Q \to \max([Q])$ voraus, was aber durch die zugrundeliegenden mathematischen Strukturen gegeben sein sollte.

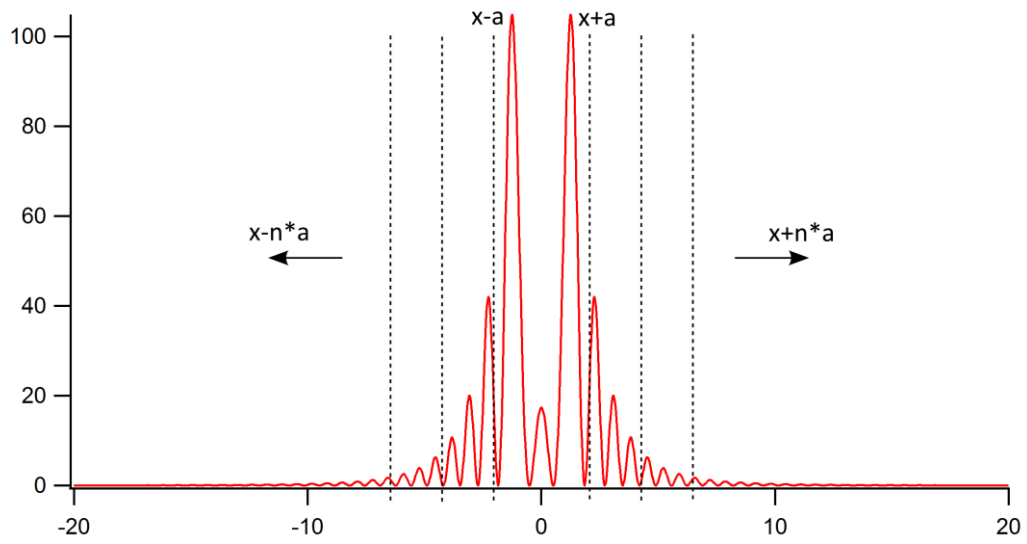

Abbildung 4.15: Schema der schrittweisen Intervallvergrößerung bei der dynamsichen Intervallbestimmung

Um ein hinreichend großes Integrationsintervall zu finden wird folgender Ansatz gewählt:

- Synthetisieren der Hilfswave in einem schmalen Bereich um *Q=0*
- Berechnung des Flächenintegrals
- Erweitern des Synthetisierungsbereichs um eine festgelegte Schrittweite (siehe Abbildung 4.15)
- Synthetisieren der Hilfswave im vergrößerten Intervall
- Berechnung des Flächenintegrals
- Vergleichen der Flächeninhalte

Dieser Vorgang wird innerhalb einer Schleife so lange fortgesetzt, bis sich die Flächeninhalte nur mehr um einen Faktor 1e-6 unterscheiden. Der Kernbereich des Algorithmus kann dem Codeausschnitt in Abbildung 4.16 zu entnommen werden.

```
//Erstellen eines temporären Wave-Objekts
make/N=500/D/FREE wTemp_sh
//Skalierung der temporären Wave
setscale x int_a,int_b,"",wTemp_sh

//Berechnung der Funktionswerte mit Multithreading
multithread wTemp_sh=standardsphere(sqrt(qy_value^2+x^2),r)
//Integration über den gesamten Skalierungsbereich
area_value_A=area(wTemp_sh)

do
    //Vergrößerung des Funktionsintervalls [a,b]
    int_a-=stepping
    int_b+=stepping
    //Re-Scaling mit vergrößertem Intervall [a-stepping,b+stepping]
    setscale x int_a,int_b,"",wTemp_sh
    //Berechnung der Funktionswerte nach Re-Scaling
    multithread wTemp_sh=standardsphere(sqrt(qy_value^2+x^2),r)
    //Integration über den gesamten Skalierungsbereich
    area_value_B=area(wTemp_sh)
    //Berechnung WaveDelta, im Verhältnis zur Ausgangswave
    area_ratio=((area_value_B-area_value_A)/area_value_A)
    //Letzten Flächenwert zwischenspeichern
    area_value_A=area_value_B
    //Abbruchbedingung wenn Flächenzuwachs kleiner 1e-6
while(area_ratio>0.000001)

return area_value_B
```

Abbildung 4.16: Berechnung der Schlitzhöhenverschmierung nach Variante 3; Codeausschnitt des Integrationsalgorithmus mit dynamischer Intervallbestimmung

Variante 2 und Variante 3 wurden mit verschiedenen Parameterkonstellationen erfolgreich auf Deckungsgleichheit geprüft. Wie erwartet, beträgt die Berechnungszeit der „Complex Variant“ ein Vielfaches der „Simple Variant“, aufgrund der langfristigen und hohen Prozessorauslastung sollte das Auswählen dieser Variante mit Bedacht gewählt werden.

Für die Synthetisierung einer Streukurve mit Schlitzhöhenverschmierung sind keine zusätzlichen Parameter notwendig, wodurch sich auch zusätzliche Eingabefelder erübrigen. Die Bedienungselemente sind im Bereich „Data synthesis – Slit-Height smearing“ untergebracht:

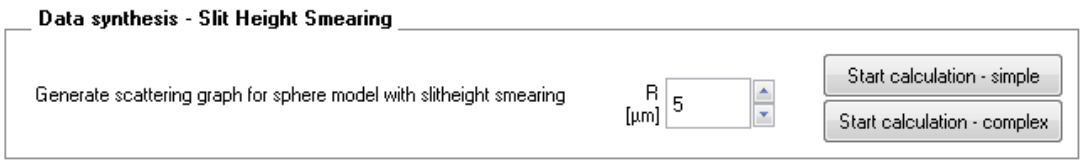

Abbildung 4.17: Oberflächenausschnitt der USANS Simulation Application für die Synthese einer schlitzhöhenverschmierten Streufunktion

Beim Betätigen des Buttons „Start calculation“ werden folgende Prozesse ausgelöst:

- Erstellen eines Wave-Objekts

- Synthetisieren der Kurvendaten durch Wave Assignment mit Multithreading: „simple“ startet Variante 2, „complex“ die langwierigere Variante 3
- Ausgabe in einem Graph-Objekt

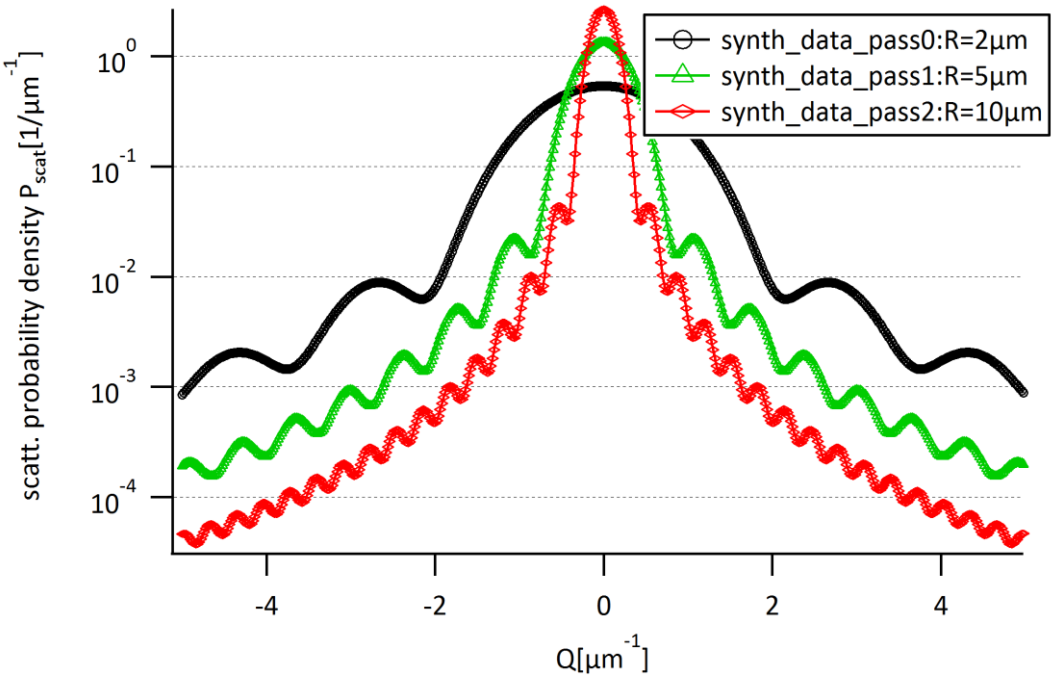

Abbildung 4.18: Synthetische Streukurven von kugelförmigen Streuobjekten, überlagert durch den Effekt der Schlitzhöhenverschmierung. Erzeugt durch Berechnungsvariante 2 – „Simple Variant“

4.2.6 Simulation der Polydispersivität

Der Einfluss einer statistischen Größenverteilung der betrachteten Streuobjekte auf die Streukurve wurde bereits im Kapitel 3.2.5.2 theoretisch behandelt. Die USANS Simulation Application bietet dementsprechend eine Möglichkeit, die „Verschmierung“ durch Polydispersivität zu simulieren. Für sphärische Geometrie wird die Streukurve durch folgende Funktion erzeugt:

$$P_{scat}(Q) = \frac{1}{C_N} \int_{R-5\sigma}^{R+5\sigma} dR \frac{1}{\sigma\sqrt{2\pi}} e^{--\frac{(R-\bar{R})^2}{2\sigma^2}} V_p^2 \left(3\frac{(\sin QR - QR\cos QR)}{(QR)^3}\right)^2 \qquad (4.7)$$

Die numerische Berechnung dieser Funktion wurde in 3 Varianten abgebildet:

Variante 1: Berechnung des indefiniten Integrals über die Integrationsvariable R mittels Mathematica. Das Ergebnis hat die Form eines Polynoms:

$$f(Q) = \frac{8e^{-2Q^2\sigma^2}\pi^2}{Q^6}\left(e^{2Q^2\sigma^2}\left(1 + Q^2(R^2 + \sigma^2)\right)\right. \\ + \left(-1 - 4Q^4\sigma^4 + Q^2(R^2 - 3\sigma^2)\right)\cos(2QR) \\ \left. - 2QR(1 + 2Q^2\sigma^2)\sin(2QR)\right) \quad (4.8)$$

In IGOR Pro kann dieses Ergebnis mittels einfachem Wave Assignment geplottet werden. In diesem Fall muss die Integration nicht durchgeführt werden, was die Gesamtzeit für die Berechnung deutlich verkürzt, deshalb handelt es sich dabei um die vergleichsmäßig performanteste Lösung. Allerdings ist diese Variante nicht ganz fehlerfrei; wie in Abbildung 4.19 ersichtlich konnten immer wieder Artefakte im Kurvenverlauf beobachtet werden, deren Auftreten bisher ungeklärt blieb.

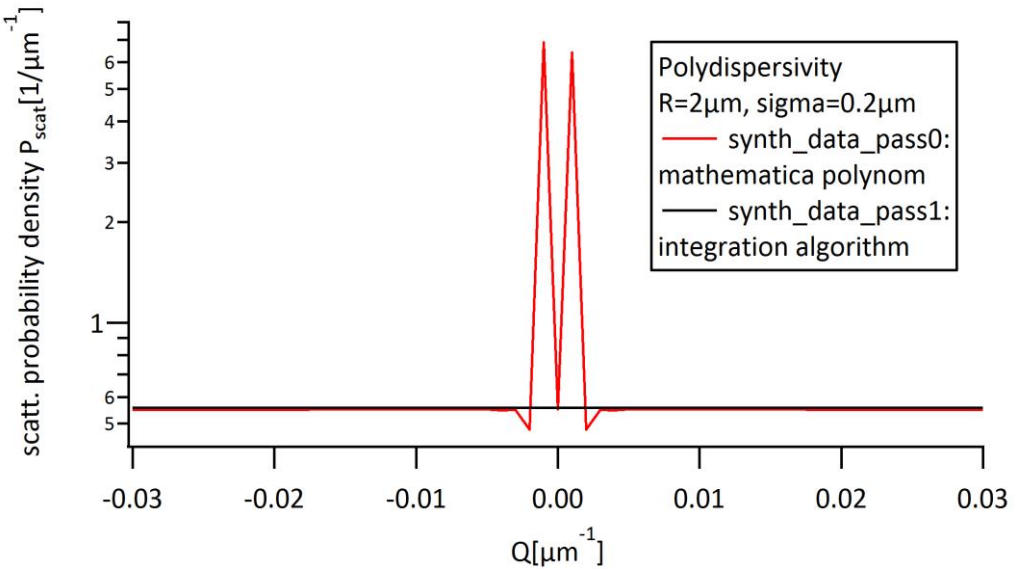

Abbildung 4.19: Stark vergrößerte Ansicht eines Artefakts im Kurvenverlauf bei Synthetisierung des Mathematica-Polynoms aus Variante 1(rot). Im Vergleich dazu, der gleichmäßige Kurvenverlauf berechnet mit Variante 3 (schwarz).

Wegen dieser Unregelmäßigkeit ist die Methode nur für Vergleichszwecke geeignet.

Variante 2: Berechnung mit Hilfe der Funktion INTEGRATE1D.

Diese Variante funktioniert fehlerfrei, ist jedoch unflexibel und nicht fähig, Berechnungen mit Multithreading durchzuführen was deutlich mehr Rechenzeit benötigt.

Variante 3: Berechnung mittels Integrationsalgorithmus.

Wie bei der Berechnung der schlitzhöhenverschmierten Streukurve in Variante 2 wird auch hier eine temporäre Hilfsmatrix zur Berechnung der Funktionswerte aufgebaut. Die Prozedur unter-

scheidet sich im Wesentlichen nur durch ihre Integrationsgrenzen und durch die zusätzliche Verwendung einer integrierten Gauß-Funktion bei der Wertezuweisung.

```
//Punkteanzahl festlegen, targetwave=Anzahl der Punkte im Graphen
    variable pts_targetwave=numpnts(targetwave)
//Punkteanzahl festlegen, helperwave=Anzahl der Punkte für die Integration
    variable pts_helperwave_poly=250
//Erstellen eines temporären 2D wave-objekts (Matrix)
    make/N=(pts_targetwave,pts_helperwave_poly)/D/FREE wTemp
//Skalierung der Matrix
    setscale x, qmin,qmax,"", wTemp
    setscale y,  r-5*sigma,r+5*sigma,"", wTemp
//Berechnung der Funktionswerte in der Matrix
    multithread wTemp=gauss(y,r,sigma)*standardsphere(x,y)
//Integration über die y-Ebene(Columns)
    integrate/DIM=1 wTemp
//Ergebnis aus der Matrix der Zielwave zuweisen
    targetwave=wTemp[p][pts_helperwave_poly]
```

Abbildung 4.20: Codeausschnitt zur Berechnung einer durch Polydispersivität verschmierten Streukurve. Ansatz mit 2-dimensionaler Wave (Matrix)

Für die Berechnung einer Streukurve mit Verschmierung durch Polydispersivität ist die Eingabe des mittleren Partikelradius R und seiner Standardabweichung σ erforderlich. Die Bedienungselemente und ein Eingabefeld für die Parameter R und σ befinden sich im Bereich „Data synthesis - Polydispersivity“.

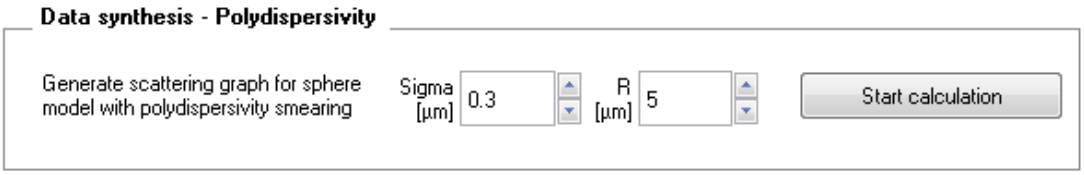

Abbildung 4.21: Oberflächenausschnitt der USANS Simulation Application für die Synthese einer durch Polydispersivität verschmierten Streufunktion

Beim Betätigen des Buttons „Start calculation“ werden folgende Prozesse ausgelöst:

- Erstellen eines Wave-Objekts
- Synthetisieren der Kurvendaten durch Wave Assignment mit Multithreading
- Ausgabe in einem Graph-Objekt

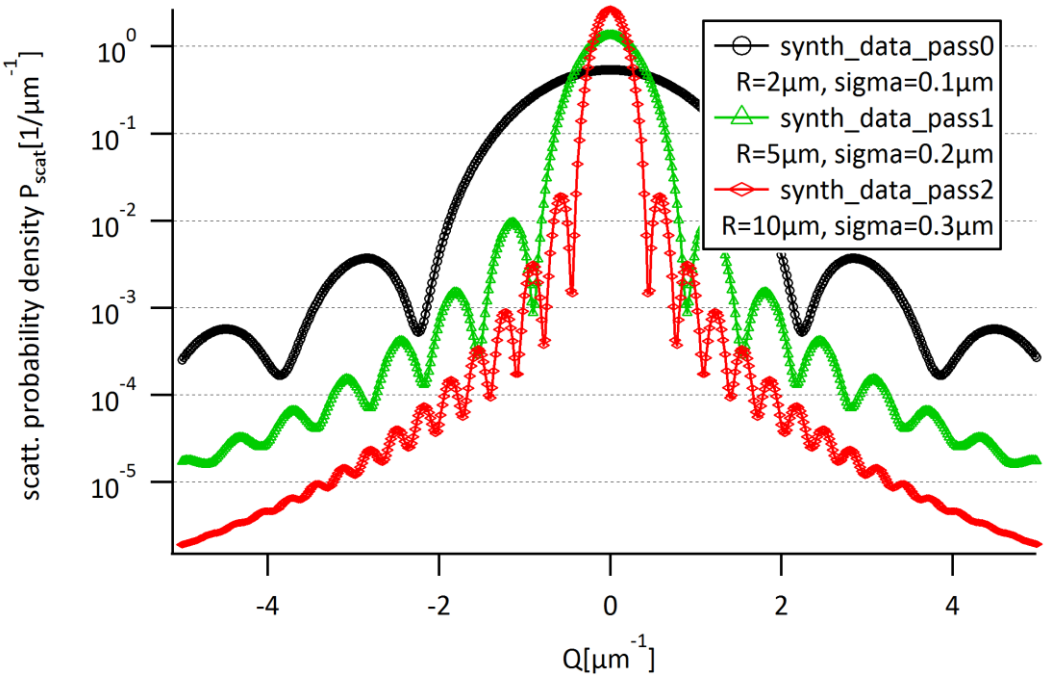

Abbildung 4.22: Synthetische Streukurven kugelförmiger Streuobjekte mit verschiedenen mittleren Radien und Standardabweichungen. Berechnung erfolgte durch Variante 3 – Integrationsalgorithmus

4.2.7 Simulation von kombinierter Schlitzhöhenverschmierung und Polydispersivität

Im Experimentablauf nimmt der Neutronenstrahl zuerst seinen Weg über die Doppelkristallanordnung, wobei die zuvor besprochene Schlitzhöhenverschmierung auftritt. Dazwischen kommt es zum Durchtritt durch die Probe, der Neutronenstrahl wird durch den Effekt der Polydispersivität weiter „verschmiert“. In der Realität überlagern sich also Schlitzhöhenverschmierungs- und Polydispersivitätseffekte. Mathematisch entspricht das der Synthetisierungsfunktion

$$P_{scat}(Q) = \frac{1}{C_N}\int_{Q_z min}^{Q_z max} dQ_z \int_{R-5\sigma}^{R+5\sigma} dR\,\frac{1}{\sigma\sqrt{2\pi}}\,e^{-\frac{(R-\bar{R})^2}{2\sigma^2}}\,V_p^2\left(3\,\frac{(\sin\hat{Q}R-\hat{Q}R\cos\hat{Q}R)}{(\hat{Q}R)^3}\right)^2 \qquad (4.9)$$

$$\hat{Q} = \sqrt{Q_y^2 + Q_z^2} \qquad (4.10)$$

wobei die Reihenfolge der Integration für das Ergebnis unbedeutend ist.

Es wurden 3 Berechnungsvarianten in IGOR Pro realisiert:

Variante 1: 2-dimensionale Integration mit der Funktion INTEGRATE1D.

Diese Variante wurde hauptsächlich für das Debugging verwendet, aufgrund der schon bekannten Nachteile und der höheren Berechnungsdauer im Vergleich zu Variante 2 ist sie aber nicht direkt über die Oberfläche der Applikation ausführbar.

Variante 2: Einfacher Integrationsalgorithmus – „Simple Variant“

Multithreading-fähiger Algorithmus, der auf dem Matrix-Ansatz aus Kapitel 4.2.5, Variante 2 aufbaut; durch geschickte Programmierung und konstante Integrationsgrenzen soll die Berechnungszeit möglichst kurz gehalten werden. Die Integration, die im Kern des Algorithmus durchgeführt wird, muss im Vergleich zur reinen Schlitzhöhenverschmierung um eine Dimension, nämlich über *R*, erweitert werden.

```
//Anzahl der Matrixpunkte festlegen
variable pts_helperwave_poly=250
variable pts_helperwave_slit=500
//Erstellen einer temporären Hilfsmatrix für die Integration
make/N=(pts_helperwave_poly,pts_helperwave_slit)/D/FREE wTemp
//Skalierung der Hilfsmatrix
setscale y, get_int_limit(r, qmin),get_int_limit(r, qmax),"", wTemp
setscale x, limit((r-5*sigma),0,r),r+5*sigma,"", wTemp
//Zuweisen der Funktionswerte
multithread wTemp=standardsphere(sqrt(qy_value^2+y^2),x)
//Integration Matrix-columns = Integration über Qz
integrate/DIM=1 wTemp
//Multipikation mit Gaussfunktion
wTemp[][pts_helperwave_slit]*=gauss(x,r,sigma)
//Integration Matrix-rows= Integration über R
integrate/DIM=0 wTemp
//Matrixergebnis dem Rückgabewert zuweisen - single value
retval=wTemp[pts_helperwave_poly][pts_helperwave_slit]
```

Abbildung 4.23: Codeausschnitt zur Berechnung einer Streukurve mit Schlitzhöhenverschmierung- und Polydispersivitätseffekten. Ansatz einfacher Integrationsalgorithmus – „Simple Variant“

Variante 3: Integrationsalgorithmus mit dynamischer Intervallbestimmung –„Complex Variant“

Multithreading-fähiger Algorithmus, der auf dem Schleifen-Ansatz aus Kapitel 4.2.5, Variante 3 aufbaut. Das ursprüngliche Konzept wird durch die Verwendung einer 2-dimensionalen Wave (Matrix) erweitert, die Integration im Algorithmus kann dadurch sehr elegant auf 2 Dimensionen, Q_z und *R*, ausgedehnt werden. Zwischen den einzelnen Integrationsschritten, die als „Zusammenfalten“ der Matrix-Wave auf einen einzelnen Zahlenwert interpretiert werden können, erfolgt die zusätzliche Gewichtung mit der für die Polydispersivität charakteristischen Gauß-Funktion. Abbildung 4.24 zeigt den wichtigsten Bereich des IGOR Programmcodes, die gesamte Funktion kann in Abschnitt 7.3 nachgeschlagen werden.

```
//Anzahl der Matrixpunkte festlegen
variable pts_helperwave_poly=250
variable pts_helperwave_slit=500
//Erstellen einer temporären Hilfsmatrix für die Integration
make/N=(pts_helperwave_poly,pts_helperwave_slit)/D/FREE wTemp
//Skalierung der Hilfsmatrix
setscale y, int_a,int_b,"", wTemp
setscale x, limit((r-5*sigma),0,r),r+5*sigma,"", wTemp
//Zuweisen der Funktionswerte
multithread wTemp=standardsphere(sqrt(qy_value^2+y^2),x)
//Integration Matrix-columns = Integration über Qz
integrate/DIM=1 wTemp
//Multipikation mit Gaussfunktion
wTemp[][pts_helperwave_slit]*=gauss(x,r,sigma)
//Integration Matrix-rows= Integration über R
integrate/DIM=0 wTemp
//Matrixergebnis dem Rückgabewert zuweisen - single value
area_value_A=wTemp[pts_helperwave_poly][pts_helperwave_slit]

do
    //Vergrößerung des Funktionsintervalls [a,b]
    int_a-=stepping
    int_b+=stepping
    //Re-Scaling mit vergrößertem Intervall [a-stepping,b+stepping]
    setscale y, int_a,int_b,"", wTemp
    setscale x, limit((r-5*sigma),0,r),r+5*sigma,"", wTemp
    //Zuweisen der Funktionswerte
    multithread wTemp=standardsphere(sqrt(qy_value^2+y^2),x)
    //Integration Matrix-columns = Integration über Qz
    integrate/DIM=1 wTemp
    //Multipikation mit Gaussfunktion
    wTemp[][pts_helperwave_slit]*=gauss(x,r,sigma)
    //Integration Matrix-rows= Integration über R
    integrate/DIM=0 wTemp
    //Matrixergebnis dem Rückgabewert zuweisen - single value
    area_value_B=wTemp[pts_helperwave_poly][pts_helperwave_slit]
    //WaveDelta, im Verhältnis zur Ausgangswave
    area_ratio=((area_value_B-area_value_A)/area_value_A)
    //letzten Integrationswert zwischenspeichern
    area_value_A=area_value_B
    //Abbruchbedingung wenn Flächenzuwachs kleiner
while(area_ratio>0.000001)

return area_value_B
```

Abbildung 4.24: Codeausschnitt zur Berechnung einer Streukurve mit Schlitzhöhenverschmierung- und Polydispersivitätseffekten. Ansatz einfacher Integrationsalgorithmus – „Complex Variant“

Anmerkung: Hat man die Konzepte der multidimensionalen Waves einmal verinnerlicht, könnte man sich an dieser Stelle die Frage stellen, warum die Streukurve nicht gleich mittels einer 3-dimensionalen Wave berechnet wird? Dieser Ansatz funktioniert prinzipiell, jedoch traten bei der

Umsetzung rasch Speicherüberläufe auf. Durch Verwendung der 64-bit Version von IGOR Pro könnte dieses Problem eventuell gelöst werden.

Für die Berechnung einer Streukurve mit Polydispersivität und Schlitzhöhenverschmierung ist die Eingabe des mittleren Partikelradius R und seiner Standardabweichung σ erforderlich. Die Bedienungselemente und ein Eingabefeld für die Parameter R und σ befinden sich im Bereich „Data synthesis – Polydispersivity & Slitheight Smearing“.

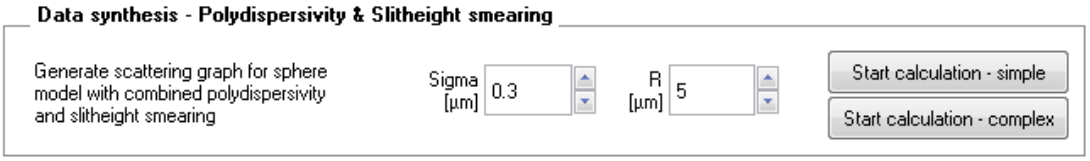

Abbildung 4.25: Ausschnitt aus der Oberfläche der USANS Simulation Application für die Synthese einer durch Schlitzhöhenverschmierung und Polydispersivität verschmierten Streufunktion

Beim Betätigen des Buttons „Start Calculation“ werden folgende Prozesse ausgelöst:

- Erstellen eines Wave-Objekts
- Synthetisieren der Kurvendaten durch Wave Assignment mit Multithreading: „simple“ für Variante 2, „complex“ für die langwierigere Variante 3
- Ausgabe in einem Graph-Objekt

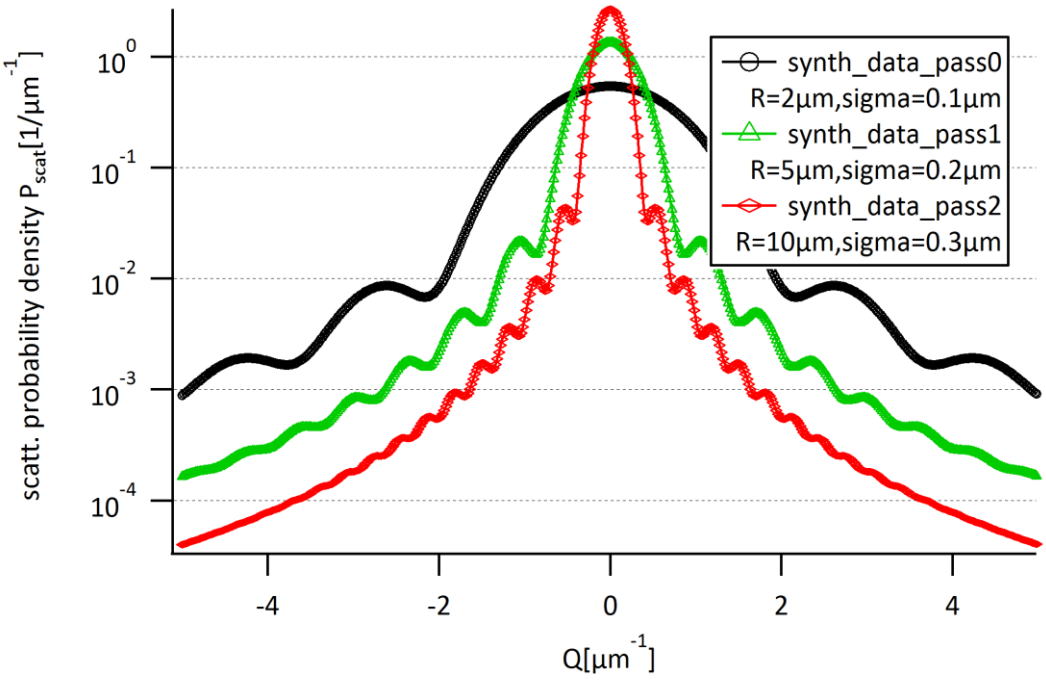

Abbildung 4.26: Synthetische Streukurven kugelförmiger Streuobjekte mit verschiedenen mittleren Radien und Standardabweichungen, zusätzlich überlagert durch Schlitzhöhenverschmierung. Berechnung erfolgte durch Variante 2 – „Simple Variant“

4.2.8 Allgemeines zur Rockingkurve, Modellfunktion

Die Leerkurve des Instruments, also eine Aufnahme des Neutronenstrahls ohne streuende Probe, wird als Instrumenten- oder Rockingkurve bzw. als Auflösungsfunktion bezeichnet. Der Begriff Rockingkurve kommt aus der Kristallographie, wo er als Synonym für Reflexionskurven von Kristallen verwendet wird. Bei neutronenoptischen Vorrichtungen, wie dem bei USANS verwendeten Doppelkristall-Diffraktometer (Bonse-Hart Kamera), kommt der Verlauf der Rockingkurve im Wesentlichen durch den kristallinen Aufbau des Monochromators und des Analysators zustande. Der genaue Aufbau dieser Vorrichtung bzw. die Entstehung und Form der realen Rockingkurve wurde bereits in vorangehenden Arbeiten behandelt, Literatur zu diesem Thema ist z.B. in [17] zu finden.

Der theoretischen Vorarbeit in [19] ist zu entnehmen, dass die Instrumentenkurve durch eine Modellfunktion

$$R(\theta) = I_0 \sum_{i=1}^{2} \left[A_t \Lambda\left(\frac{\theta - \theta_i}{\tau}\right) + A_0 \mathcal{N}(\theta_i, \sigma_0^2) + \sum_j A_j \mathcal{N}_a\left(\theta_i, \sigma_{aj}^2, \sigma_{bj}^2\right) \right] \qquad (4.11)$$

deren Parameter an Experimentaldaten gefittet werden, in sehr guter Näherung berechnet werden kann. In der USANS Simulation Application wurde dementsprechend die Möglichkeit geschaffen, aus einer Kombination von Gleichung (4.11) und den dazugehörigen Fitkoeffizienten, die Rockingkurve als Wave-Objekt zu synthetisieren (siehe Abbildung 4.27). Die Attribute dieser Rocking-Wave, wie Punkteanzahl und Intervalllänge auf der *Q*-Achse, werden beim Initialisieren speziell auf die korrespondierende Wave abgestimmt, damit es bei anschließenden Prozessen wie Faltung oder Summierung nicht zu unerwünschten Effekten kommt.

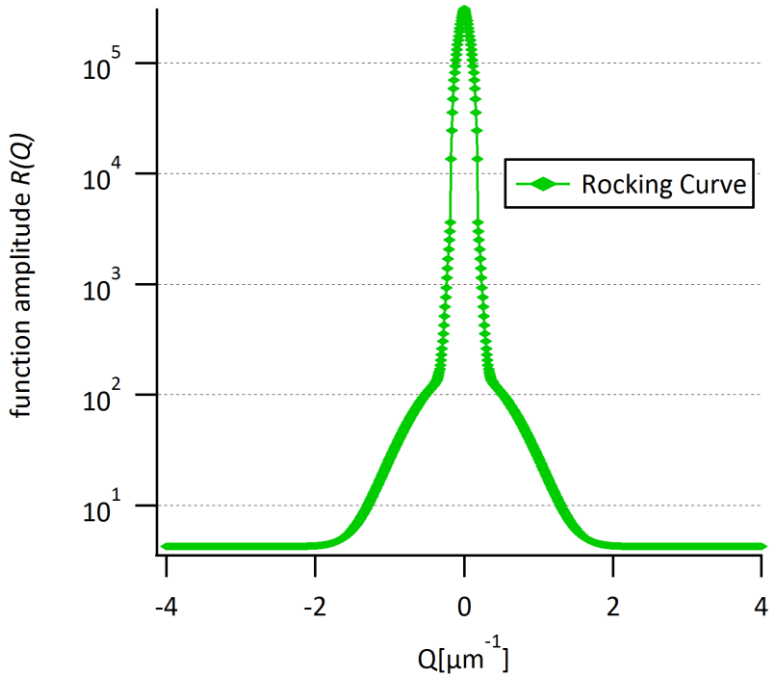

Abbildung 4.27: Synthetische Rockingkurve die mit Hilfe der Funktion S18fit berechnet wurde, in der Gleichung (4.11) implementiert ist.

4.2.9 Faltung mit der Rockingkurve

Bei der Schlitzbreitenverschmierung werden die Messdaten mit der Rockingkurve des Doppelkristall-Spektrometers verschmiert. Um diesen Prozess in die Datensynthese einfließen zu lassen, wird in IGOR Pro auf eine integrierte Funktion, die die Faltung zweier Waves, in diesem Fall einer Ausgangsfunktion und der Rockingkurve, durchführt, zurückgegriffen. Es handelt sich dabei um die Operation „Convolution“, mit der die Impulsantwort eines linearen Systems in Bezug auf ein Eingangssignal berechnet werden kann. Aus der internen Dokumentation ist zu entnehmen, dass dafür drei verschiedene Faltungstypen zur Auswahl stehen: Lineare, zirkulare und akausale Faltung.

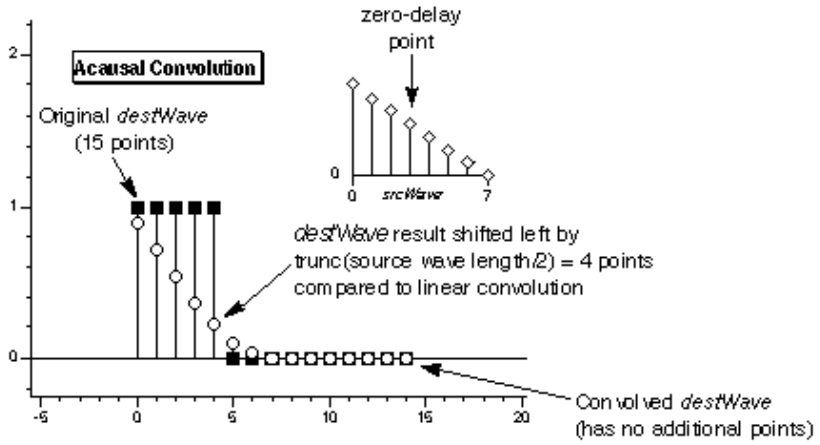

Abbildung 4.28: Eigenschaften und Beschreibung der akausalen Faltung von Waves, Auszug aus der internen Dokumentation in IGOR Pro [18]

Die akausale Option ist charakterisiert durch:

- Konstante Punkteanzahl von Ergebniswave und Ausgangswaves
- Keine Phasenverschiebung der Ergebniswave („Zero-delay“)

Im vorliegenden Anwendungsfall wurde diese Option gewählt, da die Impulsantwort, nur für diesen speziellen Faltungstyp, dieselbe Dimension und Phase wie das Eingangssignal aufweist.

Bei der Faltung einer Ausgangswave mit der Rockingkurve müssen zusätzlich folgende Punkte beachtet werden:

- Punkt- und Skalierungsübereinstimmung. Um die Faltung korrekt durchzuführen, müssen beide Waves dieselbe Punkteanzahl aufweisen, sofern auch selbe Skalierung vorausgesetzt wurde. Bei der Datensynthese wird diese Anforderung durch eine an die punktuellen Eigenschaften der Ausgangswave angepasste Erstellung der Rockingkurve erfüllt.
- Berücksichtigung des Untergrundes der Rockingkurve. In der Datensynthese wird der Untergrund der Rockingkurve vor dem Faltungsprozess ermittelt und subtrahiert. Nach der Faltung wird der Untergrund der resultierenden Ergebniskurve hinzugefügt.

Abbildung 4.29 zeigt das Ergebnis einer akausalen Faltung mit einer bereits verschmierten Streukurve und einer synthetischen Rockingkurve wie in Abbildung 4.27 dargestellt. Durch die Schlitzbreitenverschmierung kommt es zu einem weiteren „Auswaschen“ der Extrema.

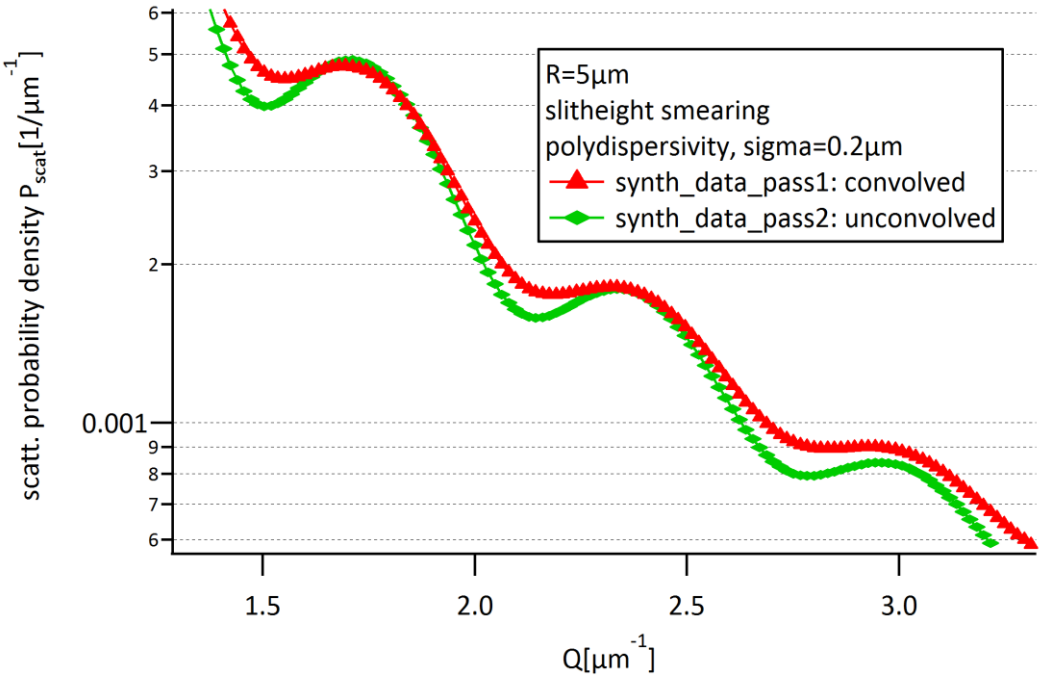

Abbildung 4.29: Darstellung des Effekts der Schlitzbreitenverschmierung, veranschaulicht anhand einer synthetischen Streukurve, deren Kurvenverlauf bereits zuvor durch Schlitzhöhenverschmierung und Polydispersivität verschmiert wurde. Es kommt zu einem zusätzlichen „Auswaschen" der Extrema.

4.2.10 Überlagerung Rockingkurve und Streukurve, Streuanteil

Beim Durchtritt des Neutronenstrahls durch die Probe ist in Abhängigkeit von Probengröße und Material nur ein gewisser Prozentanteil der Neutronen an Streuprozessen beteiligt. Die gemessene Intensität ist immer eine Überlagerung einer ungestreuten Rockingkurve und einem gestreuten Anteil, normiert auf den Gesamtneutronenfluss des einfallenden Strahls. Wird der Streuanteil über einen prozentuellen Faktor (Streuwahrscheinlichkeit) ausgedrückt, dann gilt für die Berechnung der Streukurve aus gestreutem und ungestreutem Anteil:

$$I_{ges}(Q) = (1-p)I_0(Q) + pI_s(Q), \qquad 0 \leq p < 1 \tag{4.12}$$

mit der Gesamtintensität $I_{ges}(Q)$, dem Anteil der Rockingkurve $I_0(Q)$, dem Anteil der Streukurve $I_s(Q)$ und dem prozentuellen Faktor p. In der Applikation wird die Gesamtintensität bei der Datensynthese durch die Übergabe einer Streukurve an eine dafür erstellte Funktion berechnet. Der zusätzlich benötigte, prozentuelle Faktor *p* wird über das Eingabefeld „Scattering Rate" im Oberflächenbereich „Data synthesis – USANSPOL" ausgelesen. Abbildung 4.30 zeigt den Einfluss unterschiedlicher Streuwahrscheinlichkeit auf eine Streukurve, die bereits durch Schlitzhöhen – bzw. Schlitzbreitenverschmierung und Polydispersivität verschmiert wurde. Je geringer die Streuwahr-

scheinlichkeit ausfällt, desto mehr nähert sich die synthetische Streukurve der Rockingkurve an und verliert dabei auch ihren charakteristischen Verlauf.

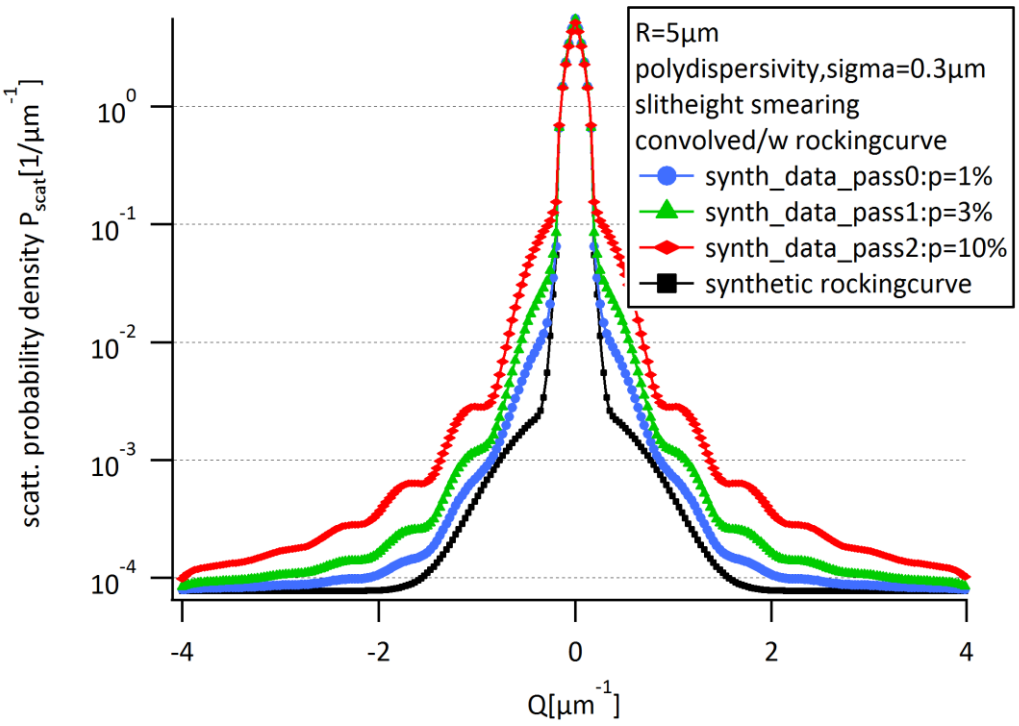

Abbildung 4.30: Einfluss unterschiedlicher Streuwahrscheinlichkeit auf eine bereits verschmierte Streukurve im Vergleich zur synthetischen Rockingkurve

4.2.11 USANSPOL Streukurven

Die letzte Ausbaustufe der Datensynthese soll auf Knopfdruck eine komplette USANSPOL Streukurve unter Einbeziehung eines umfangreichen Sets an Parametern berechnen und graphisch ausgeben. Die in den vorigen Kapiteln vorgestellten Erkenntnisse und Funktionen bilden die Grundlage für diese Simulation, die an dieser Stelle um einige magnetische und spin-abhängige Terme erweitert werden muss. Zusammengefasst soll sich die Streukurve aus folgenden Komponenten zusammensetzen:

- Sphärische Streugeometrie(Kapitel 4.2.4)
- Schlitzhöhenverschmierung – Integration über Qz(Kapitel 4.2.5)
- Schlitzbreitenverschmierung – Faltung mit Rockingkurve(Kapitel 4.2.9)
- Streuwahrscheinlichkeit(Kapitel 4.2.10)
- Polydispersivität – Gaußsche Größenverteilung der Streuobjekte(Kapitel 4.2.6)
- Polarisierter Neutronenstrahl mit Doppelpeak für Spin-up/down(Kapitel 4.2.11.4)
- Nukleare und magnetische Streulänge(Kapitel 4.2.11.1)
- Berücksichtigung des Probenwinkels(Kapitel 4.2.11.2)

4.2.11.1 Magnetische Terme

Die Aufspaltung der Streulänge in einen nuklearen und einen magnetischen Teil wird in Publikation [20] ausführlich beschrieben. Die gestreute Intensität einer bestimmten Struktur innerhalb der Probe ist gegeben durch:

$$I(Q) \propto (N\langle b(Q)\rangle V)^2 S(Q) \qquad (4.13)$$

mit der Streufunktion *S(Q)* und seinem Volumen *V*. Tritt sowohl Kernstreuung als auch magnetische Streuung auf, so ist die Streulänge gegeben durch:

$$\langle b(Q)\rangle = b_c + b_m P \cdot \widehat{M}_{\perp}(Q) \qquad (4.14)$$

Die Streulänge besteht somit aus einem nuklearen Teil, der durch die Streulänge b_c repräsentiert wird und einem magnetischen Teil, der durch das Produkt der magnetischen Streulänge b_m mit der Polarisierung *P* des Neutronenstrahls und der nuklearen Magnetisierung $\widehat{M}_{\perp}(Q)$ dessen Vektor in Richtung des Streuvektors Q gegeben ist. Weiter ist $b_m = -\gamma_n r_e S$ mit dem gyromagnetischen Verhältnis $\gamma_n = -1.913$, dem klassischen Elektronenradius $r_e = 2.818 fm$ und dem effektiven Spin *S*. Der aktuelle USANSPOL Experimentaufbau bietet die Möglichkeit, den entlang der z-Achse polarisierten, einfallenden Neutronenstrahl in eine beliebige Position in der (y,z) Ebene zu drehen, bevor er auf die Probe trifft. In vektorieller Schreibweise kann $\langle b(Q)\rangle$ entsprechend dieser Rahmenbedingungen ausgedrückt werden

$$\langle b(Q)\rangle = \langle b(Q_y, Q_z)\rangle = b_c + b_m\left[P_y \widehat{M}_{\perp y}(Q_y, Q_z) + P_z \widehat{M}_{\perp z}(Q_y, Q_z)\right] \qquad (4.15)$$

Die vektoriellen Komponenten der mittleren Magnetisierung $\widehat{M}$ sind gegeben durch:

$$\begin{aligned} \widehat{M}_{\perp y}(Q_y, Q_z) &= -\frac{Q_z}{Q^2}\left(Q_y \widehat{M}_z - Q_z \widehat{M}_y\right), \\ \widehat{M}_{\perp z}(Q_y, Q_z) &= \frac{Q_y}{Q^2}\left(Q_y \widehat{M}_z - Q_z \widehat{M}_y\right) \end{aligned} \qquad (4.16)$$

Magnetische und nukleare Streulängen liegen in der Größenordnung von [fm] und sind materialabhängig [12]:

Streulänge	Fe	Co	Ni
b_N[fm]	9.54	2.50	10.3
b_M[fm]	5.98	4.64	1.62

Um den Einfluss der magnetischen Terme möglichst anschaulich darzustellen, werden in Abbildung 4.31 die aus den vorigen Kapiteln bekannten Streukurven mit kombinierter Schlitzhöhenverschmierung und Polydispersivität verwendet. Es zeigen sich unterschiedliche Kurvenverläufe, die in erster Linie vom Verhältnis der Streulängen b_m und b_c abhängen. Steigender magnetischer Streuanteil hat einen ausgeprägten Einbruch der Streukurve um *Q=0* zur Folge.

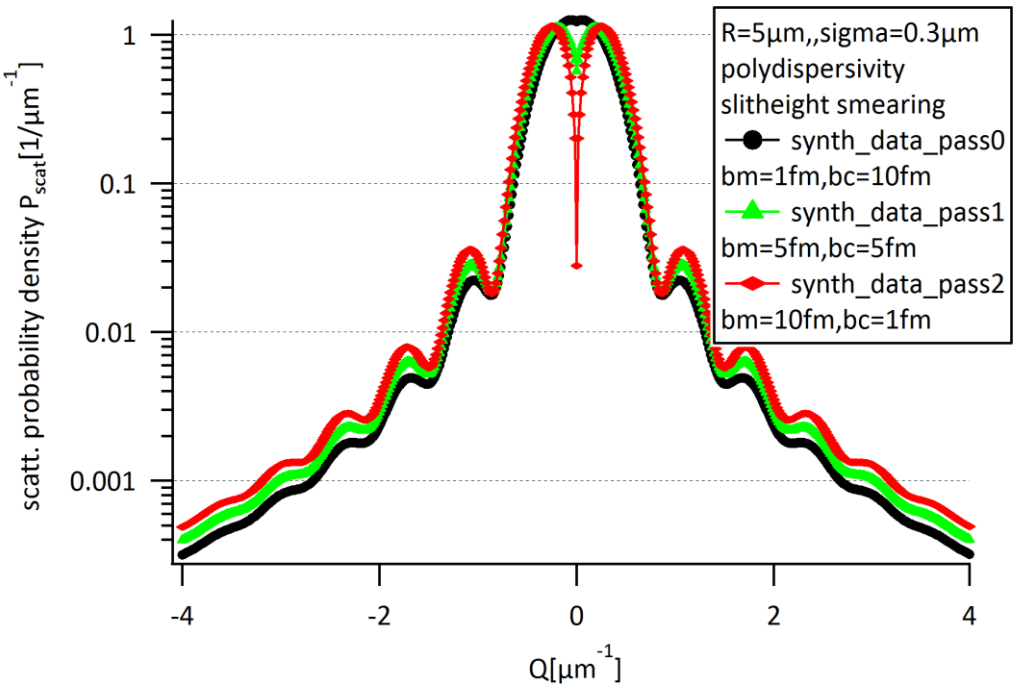

Abbildung 4.31: Synthetische Streukurven mit Schlitzhöhenverschmierung und Polydispersivität bei unterschiedlichen nuklearen und magnetischen Streuverhältnissen.

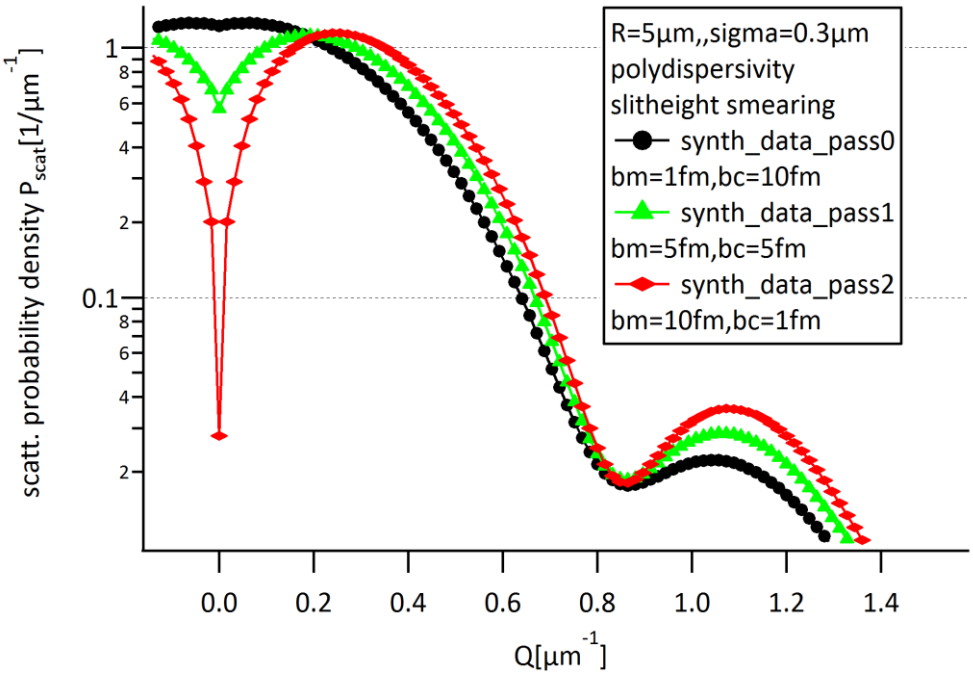

Abbildung 4.32: Detailansicht von Abbildung 4.31, Einbruch der Streukurve um Q=0 bei nuklearem und magnetischem Streuanteil

4.2.11.2 Probenwinkel und rotierter Streuvektor

Ein weiteres Feature der USANSPOL Datensynthese ist die Berücksichtigung des Winkels zwischen Streuvektor, Probenmagnetisierung und Neutronenpolarisation in Bezug auf eine festgelegte Probenachse. Die Begründung dafür liegt in der Annahme, dass durch Drehung der Probe und anschließende Messung in verschiedenen Winkeln zusätzliche Informationen für die Datenanalyse erlangt werden können, ähnlich einer tomographischen Rekonstruktion. Bei diesem Verfahren wird die Richtung des Streuvektors, der Probenmagnetisierung und der Neutronenpolarisation variiert, während die innere Struktur der Probe unverändert bleibt. In die Berechnungen fließt diese Winkelabhängigkeit durch einen richtungsabhängigen Streuvektor ein, es gilt für die rotierten Vektoren Q_y' und Q_z'

$$Q_y'(\theta) = Q_y \cos(\theta) + Q_z \sin(\theta)\,, \qquad Q_z'(\theta) = Q_z \cos(\theta) - Q_y \sin(\theta) \qquad (4.17)$$

Abbildung 4.33 zeigt, wie sich der Verlauf der Streukurve verändert, wenn die Probe bei fixierter Polarisation und Magnetfeldausrichtung gedreht wird. Ausgangspunkt der Berechnung ist eine verschmierte Streukurve mit magnetischem Anteil, Polarisation und Magnetfeld wurden auf 0° fixiert, was definitionsgemäß der Richtung senkrecht nach oben entspricht. Bei einer Probenausrichtung normal zu *P* und *M* zeigt sich eine phänomenologisch unterschiedliche Kurvenform.

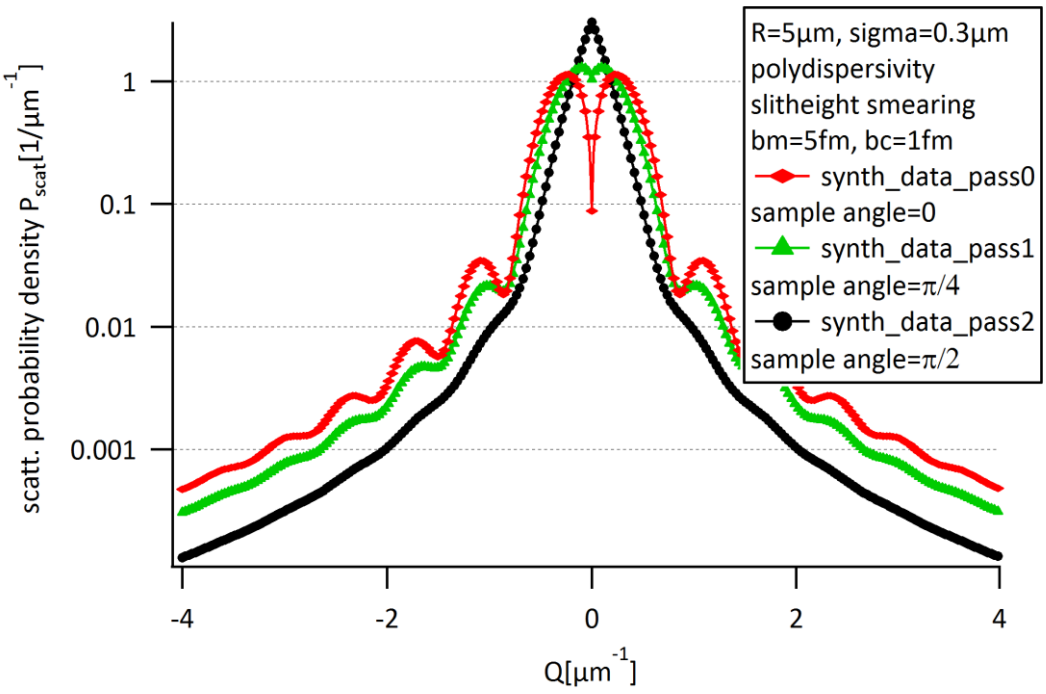

Abbildung 4.33: Einfluss der Probenausrichtung auf verschmierte Streukurven mit nuklearem und magnetischem Streuanteil. Während der Berechnung für die Probenwinkel 0, π/4 und π/2 wurden Polarisation und Magnetfeldausrichtung auf 0° fixiert.

Anmerkung: Angelehnt an die Verfahrensweise im Experiment werden in der Applikation bei einer Änderung des Probenwinkels automatisch die Polarisierung *P* und die Magnetisierung *M* um denselben Betrag gedreht. Dieses Verhalten kann bei Bedarf manuell überschrieben werden.

4.2.11.3 Synthetisierungsfunktion

Erweitert man nun die Funktion der kombinierten Schlitzhöhenverschmierung und Polydispersivität um die oben beschriebenen magnetischen Komponenten und den rotationsabhängigen Streuvektor, so erhält man für die USANSPOL Synthetisierungsfunktion

$$P_{scat}(Q) = \frac{1}{C_N} \int_{Q_z min}^{Q_z max} dQ_z \int_{R-5\sigma}^{R+5\sigma} dR \left(N\langle b(Q'(\theta))\rangle V(R) \right)^2 f(R)\, F(Q'(\theta), R) \qquad (4.18)$$

$$f(R) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(R-\bar{R})^2}{2\sigma^2}} \qquad (4.19)$$

$$F(Q'(\theta), R) = V_p^2 \left(3 \frac{(\sin Q'(\theta)R - Q'(\theta)R\cos Q'(\theta)R)}{(Q'(\theta)R)^3} \right)^2 \qquad (4.20)$$

$$\langle b(Q'(\theta))\rangle = b_c + b_m P \cdot \widehat{M}_{\perp}(Q'(\theta)) \quad (4.21)$$

$$Q'(\theta) = \sqrt{Q'_y(\theta)^2 + Q'_z(\theta)^2} \quad (4.22)$$

4.2.11.4 Doppelpeaks und Spinzustände

Bei USANSPOL Experimenten wird der einfallende Neutronenstrahl mittels Magnetprismen aufgespalten [10], er besteht danach aus zwei Komponenten entsprechend den Spinzuständen Up und Down. Im Streubild sind daher die für USANSPOL charakteristischen, um den Punkt *Q=0* verschobenen Doppelpeaks zu erkennen, der Betrag ihrer Verschiebung auf der *Q*-Achse ist eine instrumentenabhängige Größe und wird hier nicht weiter behandelt. In der Applikation wird jedenfalls von einer symmetrischen Verschiebung ausgegangen. Weiters resultiert aus Gleichung (4.15), dass die Maxima der beiden Peaks für Spin-up und Spin-down aufgrund ihrer, dem Vorzeichen nach unterschiedlichen magnetischen Streuanteile nicht dieselben Intensitäten aufweisen. Im letzten Schritt der Datensynthese werden die beiden separat berechneten Streukurven für Spin-up und Spin-down um den Betrag des Parameters Peakshift verschoben und anschließend betragsmäßig zur Gesamtkurve zusammengesetzt.

4.2.11.5 Implementierung

Wie schon in der Kapiteln zuvor ist die USANSPOL-Datensynthese aufgrund der unterschiedlichen Herangehensweise bei der Einbeziehung von Schlitzhöhenverschmierung in eine einfache und eine komplexe Berechnungsmethode mit dynamischer Intervallbestimmung unterteilt. Ausgehend von den Methoden für die Berechnung der kombinierten Schlitzhöhenverschmierung und Polydispersivität müssen die Prozeduren für USANSPOL Streukurven um folgende Funktionen erweitert werden:

- Magnetische Terme
- Probenrotation/Streuvektor
- Faltung mit der Rockingkurve
- Streuanteil/Streuwahrscheinlichkeit

- Berechnung einzelner Streukurven mit verschobenen Peaks für beide Spinzustände und anschließende Zusammensetzung zur Gesamtkurve

Für die Berechnung einer kompletten USANSPOL Streukurve ist die Eingabe eines ganzen Sets an Parametern erforderlich. Die Bedienungselemente für die Erstellung und die Eingabefelder für die Parameter befinden sich im Bereich „Data synthesis - USANSPOL“.

Aufgrund der Abmessungen des zugehörigen Bereichs auf der Oberfläche der Applikation wird hier auf eine Abbildung verzichtet und auf Kapitel 4.2.13, das die gesamte Oberfläche zeigt, verwiesen.

Abbildung 4.34 zeigt das Resultat der „finalen Ausbaustufe“ der USANS Simulation Application: Eine numerisch berechnete USANSPOL Streukurve auf Basis des „Default Parameter Set“ (siehe Kapitel 7.2) im Vergleich zur synthetischen Rockingkurve.

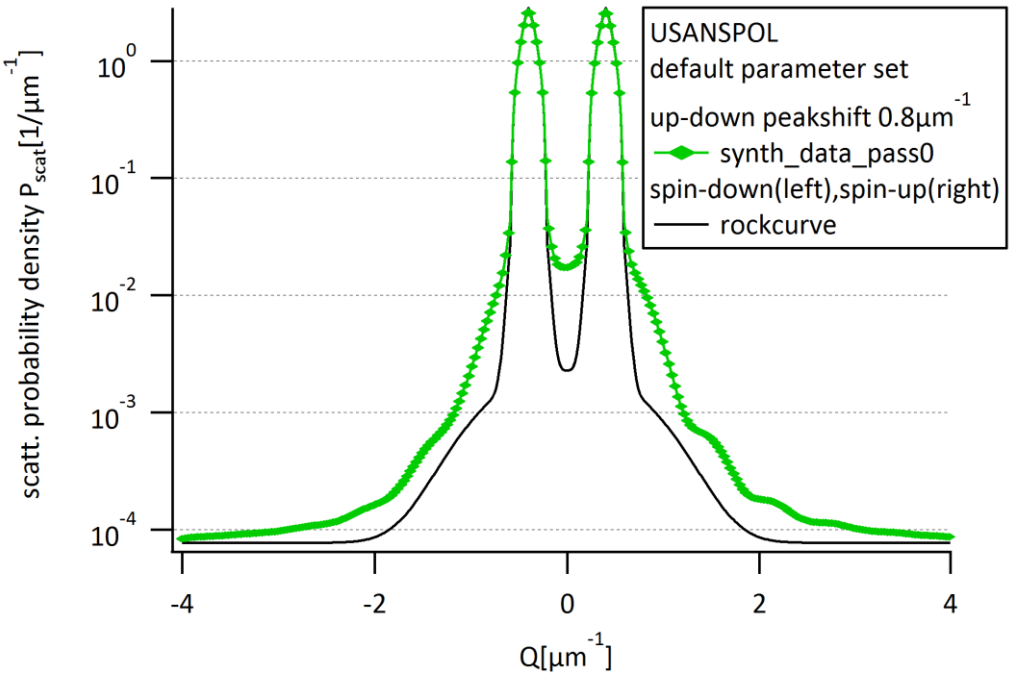

Abbildung 4.34: USANSPOL Streukurve, berechnet mit dem Default-Parameterset und synthetische Rockingkurve mit Doppelpeak im Vergleich

4.2.12 Überlagerung der Streudaten mit statistischem Rauschen

Um das Verhalten realer Messdaten, deren Verteilung im Normalfall durch ein statistisches „Rauschen“ gekennzeichnet ist, nachzustellen, wurde in der Applikation die Möglichkeit geschaffen, synthetisierte Waves mit einem ähnlichen Muster zu versehen. Programmtechnisch wurde dafür die integrierte Funktion GNOISE eingesetzt, mit der Zufallswerte einer Gauß-Verteilung in einer

Weise erzeugt werden können, sodass die Standardabweichung einer unendlichen Anzahl dieser Werte gegen einen vorgegebenen Wert konvergiert. Durch Addieren dieser sowohl positiven als auch negativen Zufallswerte zu den Funktionswerten von synthetischen Streukurven entsteht ein Streubild, das sich dem experimentell Beobachteten durchaus annähert.

$$P_{scat}(Q) = \frac{1}{C_N}\left[\, I(Q) + gnoise\left(\sqrt{I(Q)}\right)\right] \qquad (4.23)$$

In Anlehnung an Abbildung 4.34 zeigt Abbildung 4.35, wie sich der Verlauf der USANSPOL Streukurve verändert, wenn zusätzlich statistisches Rauschen in Form von Gleichung (4.23) in die Berechnung mit einfließt.

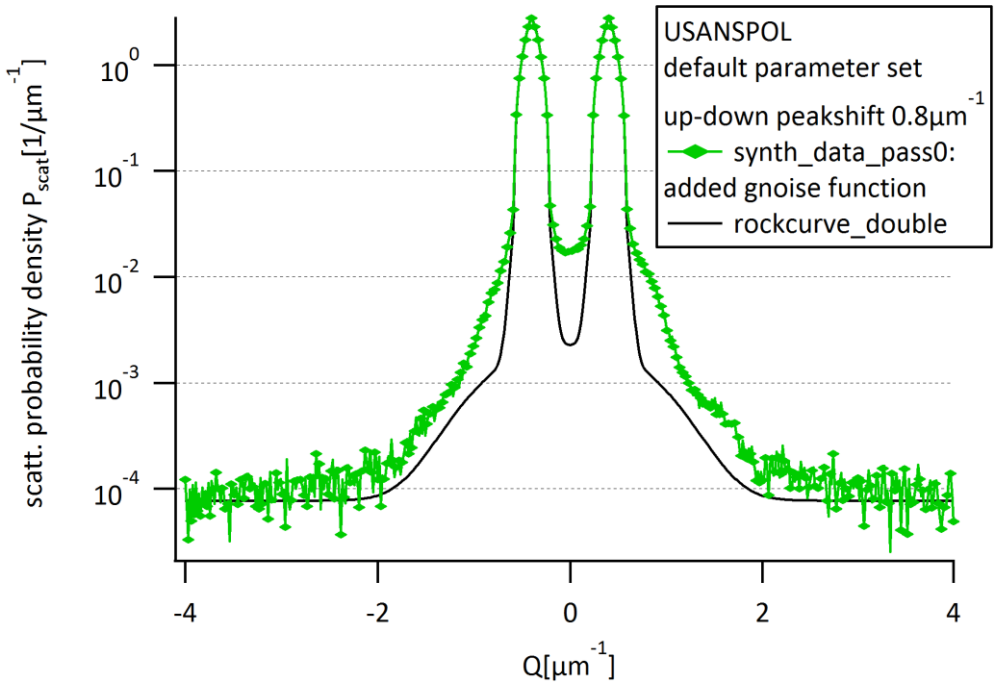

Abbildung 4.35: USANSPOL Streukurve, berechnet mit dem Default-Parameterset und zusätzlichem statistischem Rauschen (GNOISE), sowie die synthetische Rockingkurve. Die relative Stärke des Rauschens steigt mit der geringer werdenden Intensität proportional zum Betrag des Streuvektors.

Je höher die Zählrate einer experimentell gemessenen Streukurve ist, desto geringer fällt die statistische Schwankungsbreite für jeden Funktionswert von Q aus; da die Intensität mit steigendem Betrag von *Q* stark abnimmt, ist in den äußeren Bereichen der Kurve deutlich mehr Rauschen sichtbar.

Zusätzlich bietet die USANS Simulation Application durch den Parameter „Intensity Equivalent [cts]“ die Möglichkeit, bei der Berechnung des Rauschpegels einen variablen Wert für die Gesamt-

zahl der gemessenen Neutronen (äquivalent zu den Counts in einem Experiment) anzusetzen. Der Höhe des Wertes kann als Maß für die Stärke des Rauschens verstanden werden: niedrige Werte produzieren Streukurven mit starken statistischen Schwankungen, höhere Werte führen zu einem „ruhigeren“ Kurvenverlauf.

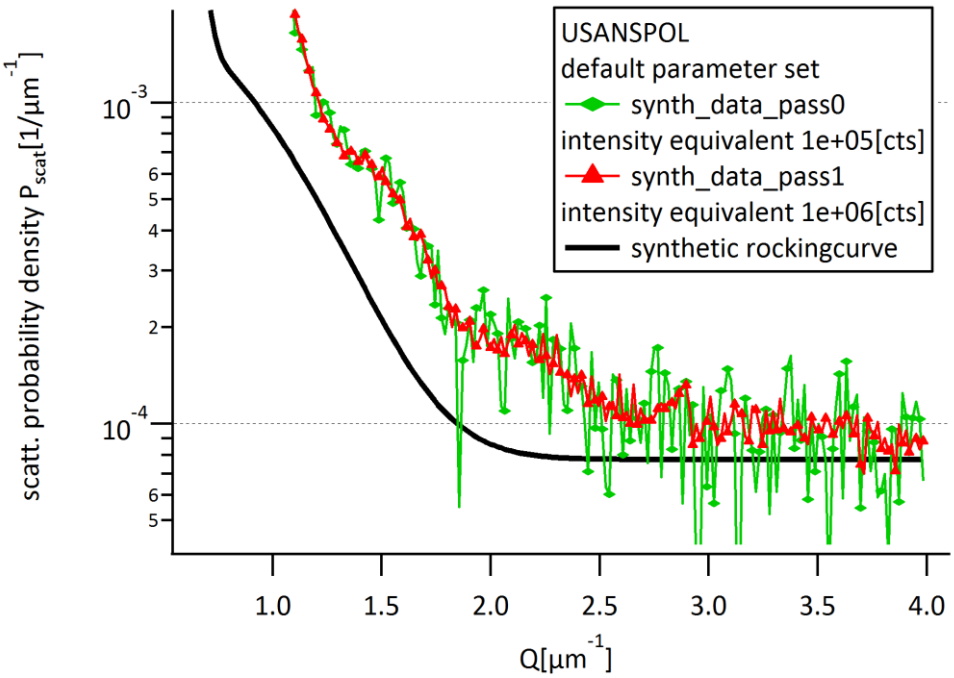

Abbildung 4.36: Berechnete USANSPOL Streukurven mit Default-Parameterset und unterschiedlich starkem gaußschem Rauschen, dass von zwei verschiedenen Intensitäts-Niveaus (Intensity Equivalent) ausgeht sowie die synthetische Rockingkurve als Anhaltspunkt.

4.2.13 Oberfläche der Applikation

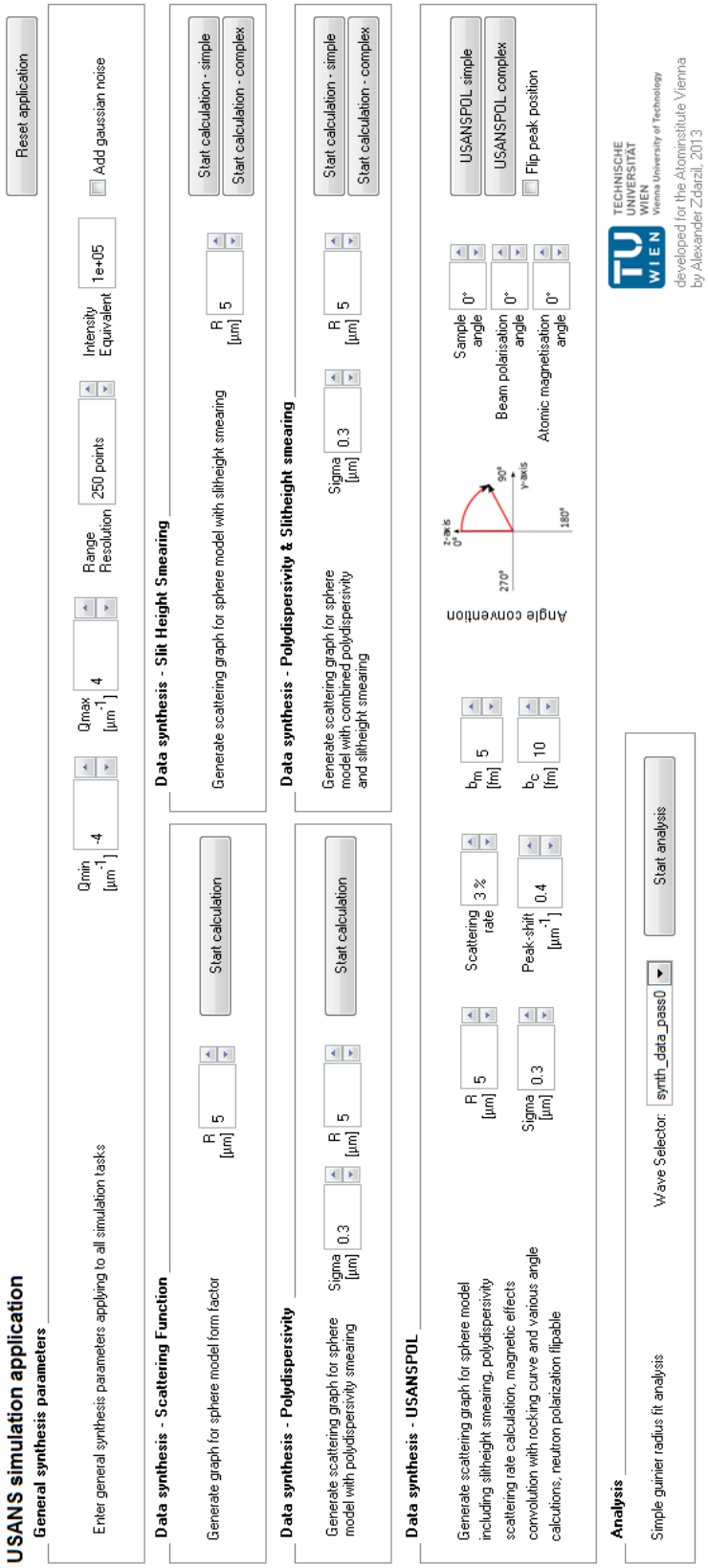

Abbildung 4.37: USANS Simulation Application – Schnappschuss der Oberfläche nach dem Initialisieren, d.h. im Status des Default Parameter Set. Der Quellcode der darauf abgebildeten IGOR-Objekte kann dem Abschnitt 7.3 ab Seite 116 entnommen werden.

4.2.14 Applikationstest

Application Testing ist ein selbstverständlicher Teil der Entwicklung eines Programms und sollte nie vernachlässigt werden. Im Falle der USANS Simulation Application wurden

- Tests in der Entwicklungsphase
- einige selbst erstellte Testfälle
- Begutachtung und Test durch den Betreuer

durchgeführt. Es sei darauf hingewiesen, dass kein systematisches Testing mit einer großen Anzahl an Testfällen und anschließender Auswertung durchgeführt wurde.

4.2.15 Ausblick

Im Moment ist die **Auswahl der Teilchengeometrien** auf sphärisch symmetrische Objekte beschränkt. Wie in Kapitel 3.2.2 gezeigt wurde, ist es möglich Formfaktoren bzw. Formfunktionen analytisch zu berechnen, in diversen Quellen, wie z.B. in [8] wird eine große Auswahl an Geometrien wie z.B. elliptisch, quaderförmig, streifenförmig beschrieben. Diese Funktionen könnten modular in die Applikation integriert werden, um dem User anschließend über geeignete Bedienungselemente die Möglichkeit zu bieten, Berechnungen mit unterschiedlichen Teilchenformen anzustoßen. Gerade in Kombination mit der Möglichkeit den Probenwinkel festzulegen (was bezüglich nuklearer Streuung bei sphärisch symmetriescher Geometrie vernachlässigbar ist), würden sich bei rotations-asymmetrischen Strukturen interessante Effekte zeigen.

Wenn die Applikation von einer breiteren Personengruppe eingesetzt werden soll, etwa im Rahmen des Lehrbetriebes, wäre es sinnvoll, die **Programmierung in gewissen Bereichen robuster** zu gestalten. Abhängig vom jeweiligen Programmbereich gäbe es sicher Potential, den Umgang mit der Anwendung durch das Hinzufügen von vermehrten Prüf- und Ausnahmeroutinen (z.B. im Bereich der Interaktion des Users mit der Oberfläche) zu verbessern.

Eine Möglichkeit, die Simulation näher an die Realität zu bringen, wäre, bei der Berechnung des Formfaktors eine Stufe „früher“ anzusetzen; aus Gleichung (3.27) wird ersichtlich: Der **Formfaktor entspricht der Fourier-Transformierten** der Formfunktion $s(\vec{r})$. Damit wäre es vorstellbar, den Formfaktor, nicht wie in dieser Arbeit gezeigt, zuvor analytisch zu bestimmen, sondern nach Definition der Formfunktion numerisch zu berechnen. Auch für diesen Fall hätte IGOR Pro schon die richtigen Werkzeuge in Form von FFT (Fast Fourier Transform) Algorithmen mit an Bord.

5 Auswertungsbeispiele

Im Anschluss werden einige demonstrative Beispiele angeführt, die veranschaulichen, wie durch Verwendung der USANS Simulation Application schnell interessante wissenschaftliche Zusammenhänge untersucht werden können. Die Anwendung wurde genau zu diesem Zweck mit einem umfangreichen Set an Eingabeparametern ausgestattet; ihre Variation bzw. die resultierenden Ergebnisse ermöglichen es, eine Vielzahl an Fragestellungen zu behandeln.

5.1 Gegenüberstellung der Verschmierungsarten

Die im Kapitel 4.2.4 vorgestellte Streufunktion und ihre in 4.2.5, 4.2.6 und 4.2.7 behandelten Verschmierungsarten werden bezüglich einer festgelegten Objektgröße direkt gegenübergestellt. In Abbildung 5.1 zeigt sich nicht nur das bereits erwähnte Verschmieren der Kurvenextrema, sondern auch die erhöhte Wahrscheinlichkeitsdichte im Kurvenverlauf einer schlitzhöhenverschmierten Intensität.

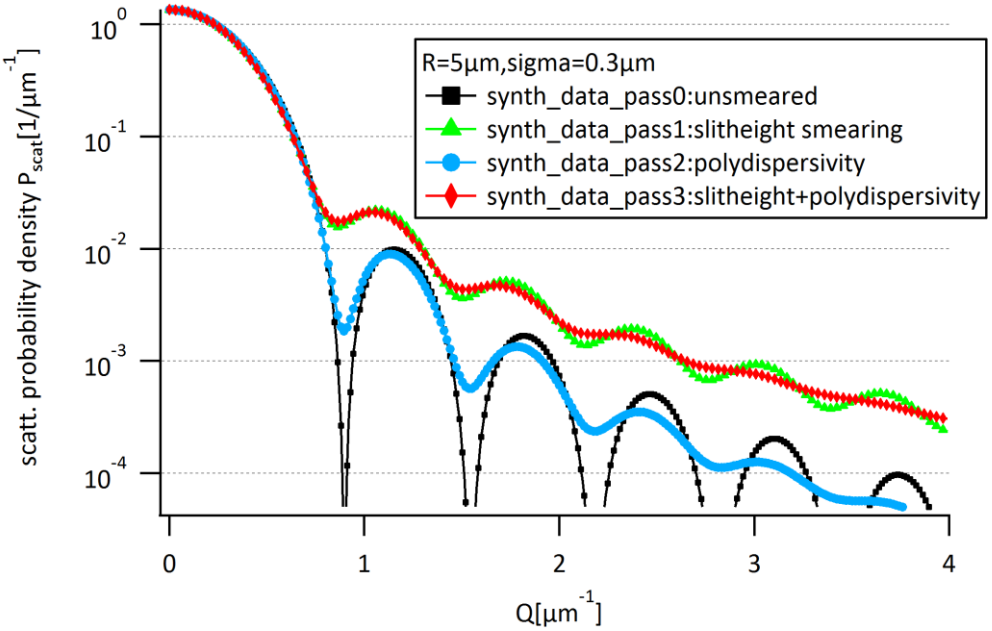

Abbildung 5.1: Direkte Gegenüberstellung verschiedener Verschmierungsarten bei gleichbleibender Objektgröße. Deutlich zu erkennen ist das „Ausschmieren" der Extrema der Streukurve, sowie die erhöhte Wahrscheinlichkeitsdichte bei Anwesenheit von Schlitzhöhenverschmierung.

5.2 USANSPOL – Einfluss der Streuwahrscheinlichkeit

Analog zu Abbildung 4.30 (Singlepeak) soll der Einfluss der Streuwahrscheinlichkeit auf die finale USANSPOL Streukurve untersucht werden. Abbildung 5.2 zeigt das Ergebnis dieser Auswertung, seine Interpretation weicht im Prinzip nicht von jener in Abschnitt 4.2.10 ab. Erwartungsgemäß nähert sich die Streukurve bei geringem Streuanteil immer mehr der Rockingkurve an.

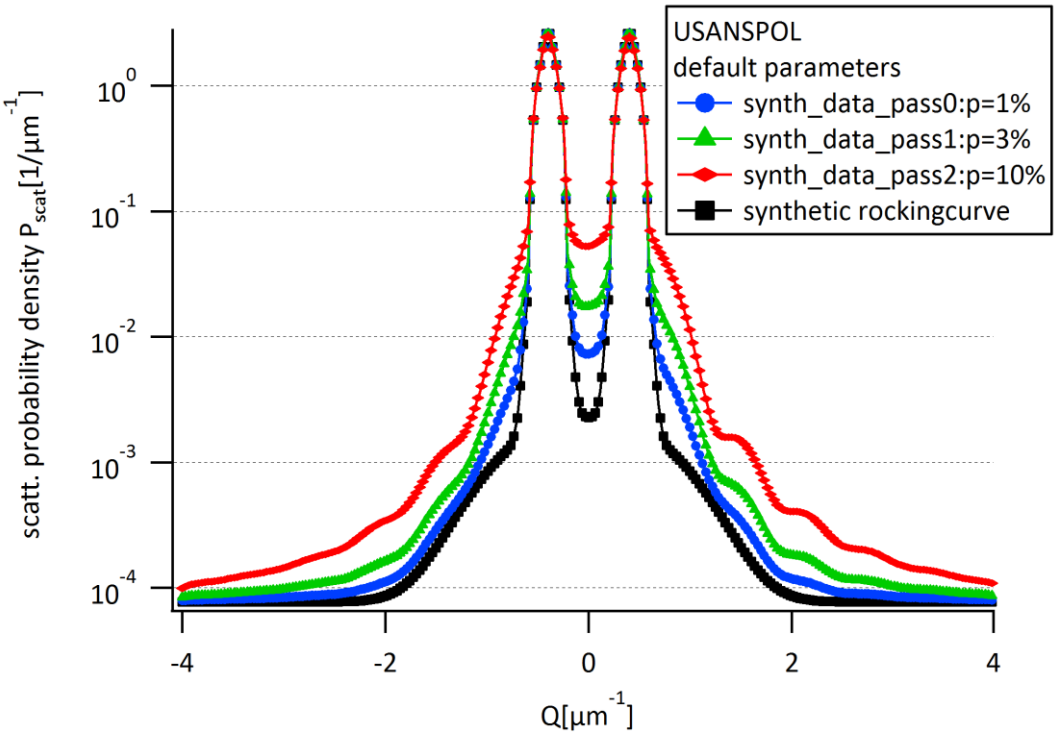

Abbildung 5.2: Einfluss der Streuwahrscheinlichkeit auf USANSPOL Streukurven, Default-Parameter Set. Mit abnehmender Streuwahrscheinlichkeit verschwinden die charakteristischen Kurvenzüge, deren Verlauf durch die Struktur des Probenmaterials verursacht wird.

In den Arbeiten von Rechberger [21] und Mach [22] werden durch den Einsatz von neu entwickelten, analytischen Verfahren Streuwahrscheinlichkeiten im Bereich von ~20-50% beschrieben. Obwohl anzunehmen ist, dass das in der vorliegenden Arbeit erhaltene Ergebnis, aufgrund eines unterschiedlichen Zugangs nicht direkt mit den oben genannten vergleichbar sein wird, soll dieser gefühlsmäßig hohe Streuwahrscheinlichkeitsbereich trotzdem vom Blickwinkel der Datensynthese aus beleuchtet werden. Abbildung 5.3 zeigt das Ergebnis einer Simulation dieser Fragestellung mit der USANS Simulation Application: Für Streukurven mit hohem Streuanteil ist die Amplitude der Wahrscheinlichkeitsdichte in den Bereichen der Peaks verringert, während der Verlauf der beiden Kurvenflanken, im Vergleich zu höheren Werten hin verschoben ist.

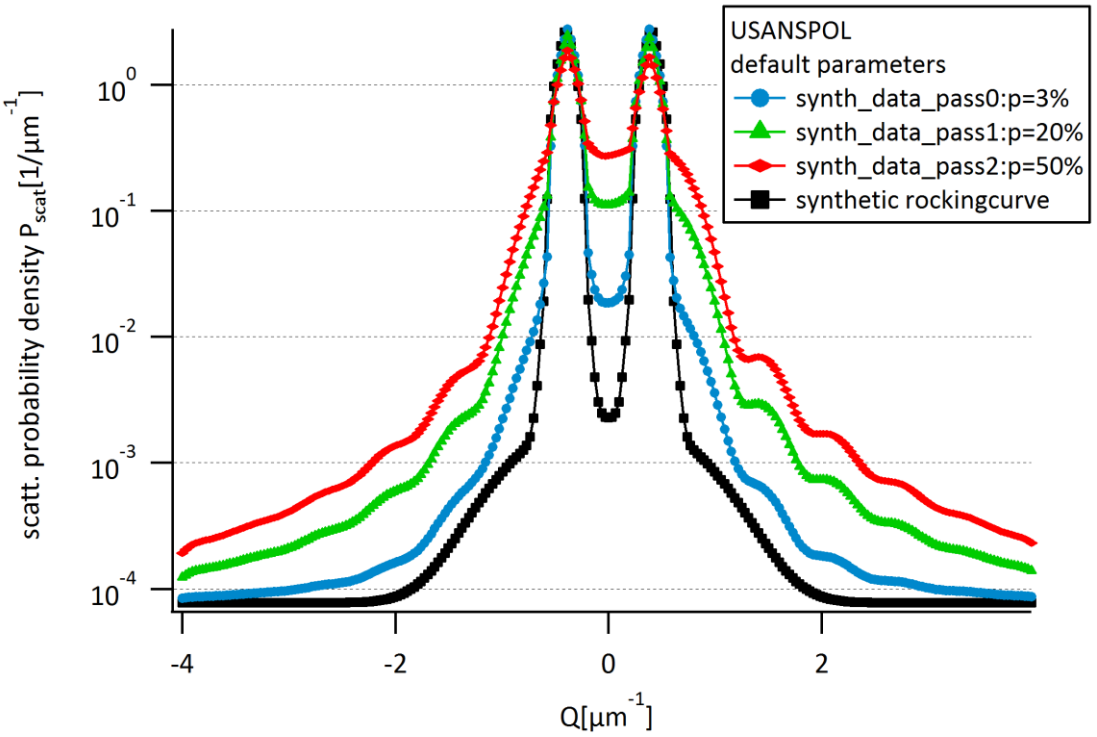

Abbildung 5.3: Synthetische USANSPOL Streukurven (Default Parameter Set) und Rockingkurve bei hohen Streuwahrscheinlichkeiten. Mit zunehmendem Streuanteil verliert die Rockingkurve deutlich an Einfluss.

5.3 USANSPOL – Unterscheidbarkeit von verschmierenden Effekten

Im Prinzip gibt es vier Arten von Verschmierungen, die auf eine ideale Streukurve wirken können: Schlitzbreiten- und höhenverschmierung, Polydispersivität und der Verlust von charakteristischen Kurvenverläufen durch den Streuanteil zwischen Streukurve und Rockingkurve, der ebenfalls als Verschmierung gesehen werden kann. Von diesen vier Arten sind, eine konstante Rockingkurve vorausgesetzt, nur die letzten zwei parametrisierbar. Eine Auswertung mit der USANS Simulation Application soll klären, ob durch Variation der Parameter „Sigma“ (Polydispersivität) und „Scattering rate“ (Streuanteil) ähnliche Ergebnisse erzielt werden können, oder besser gesagt, ob die beiden Verschmierungsarten unterscheidbar bleiben.

Um diesen Vergleich genauer und vor allem isoliert beurteilen zu können, wird in den nachfolgenden Berechnungen rein nukleare Streuung angesetzt.

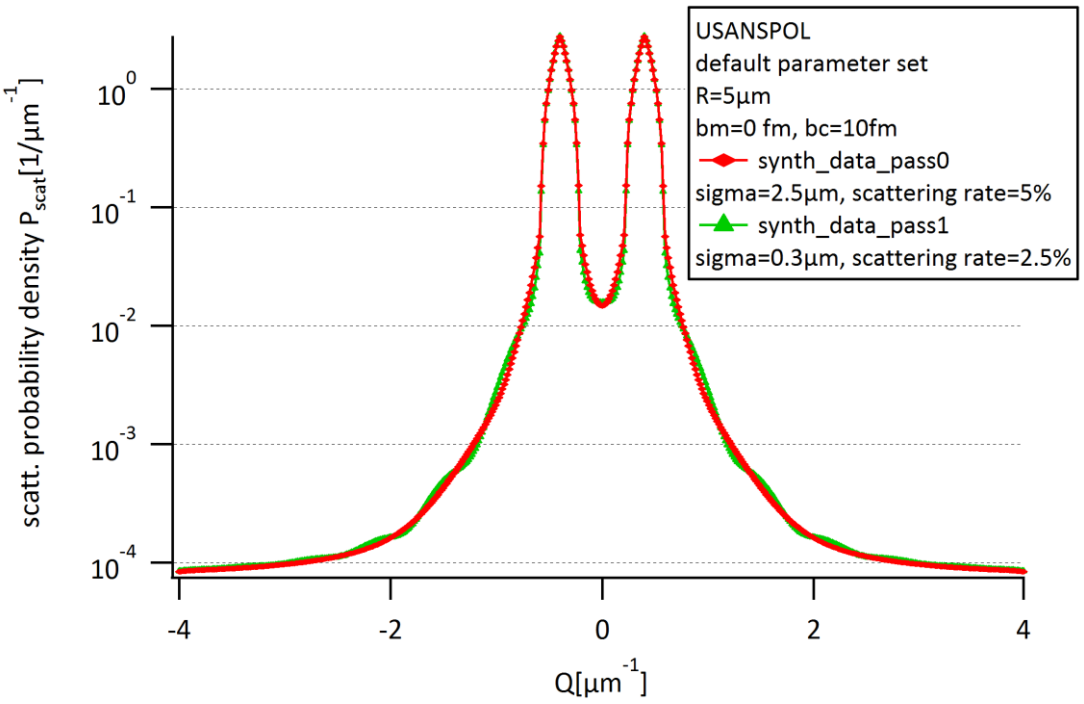

Abbildung 5.4: USANSPOL Streukurve bei rein nuklearer Streuung (Objektgröße R =5 µm) unter Variation der Parameters Sigma und der Scattering rate. Durch geeignete Wahl konnten sehr ähnliche Kurven für diesen Streuer erzeugt werden.

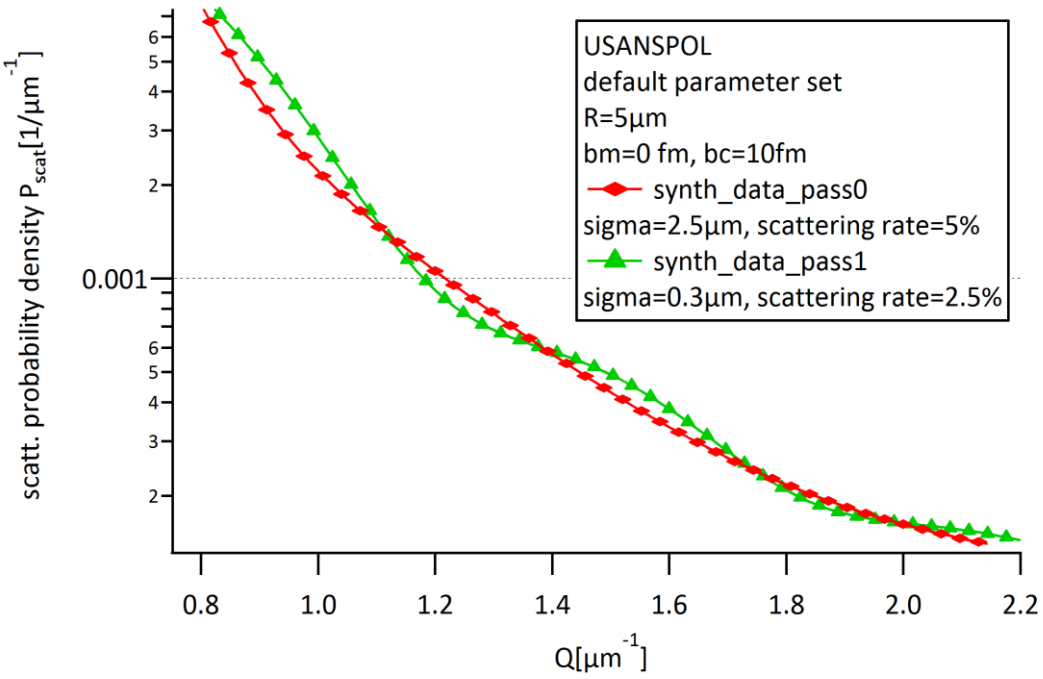

Abbildung 5.5: Detailansicht der USANSPOL Streukurve aus Abbildung 5.4. Durchaus ähnliche Kurvenverläufe, trotz geringerer Scattering rate ist der charakteristische Kurvenverlauf bei synth_data_pass1 noch zu erkennen.

Da die Streukurve bei größerem Objektradius erfahrungsgemäß einen glatteren Verlauf nimmt, wird der Vergleich für doppelten Radius R=10 µm wiederholt.

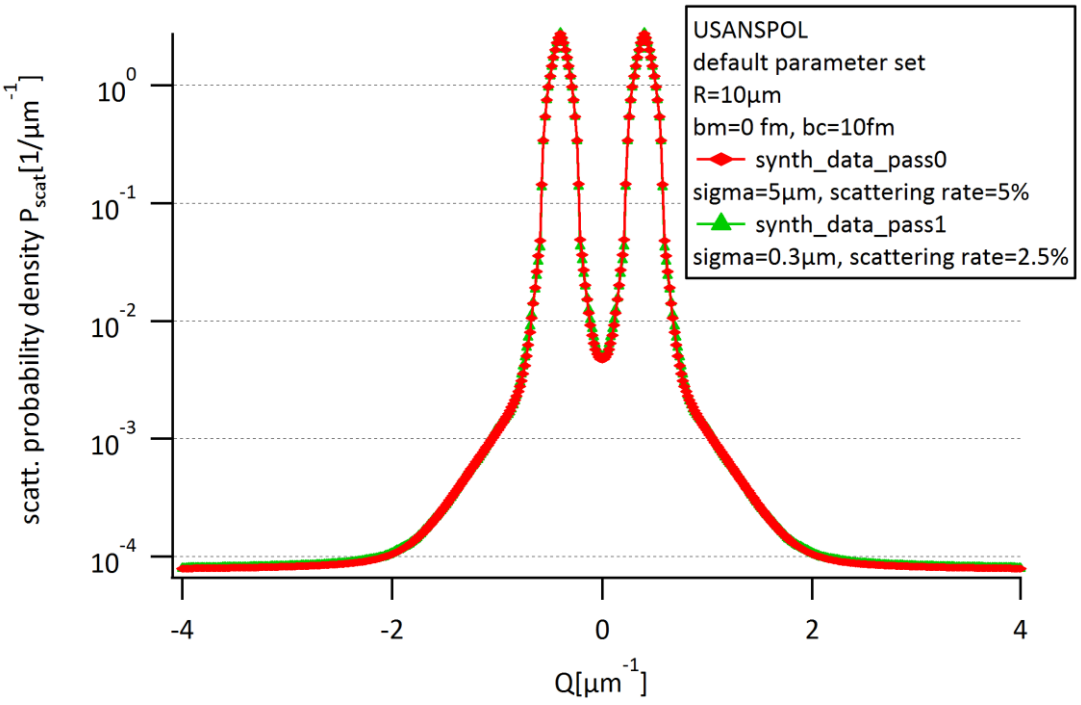

Abbildung 5.6: USANSPOL Streukurve bei rein nuklearer Streuung (Objektgröße R = 10 µm) unter Variation der Parameter sigma und scattering rate. Durch geeignete Parameterwahl konnten Kurven erzeugt werden, die bei dieser Skalierung kaum unterscheidbar sind.

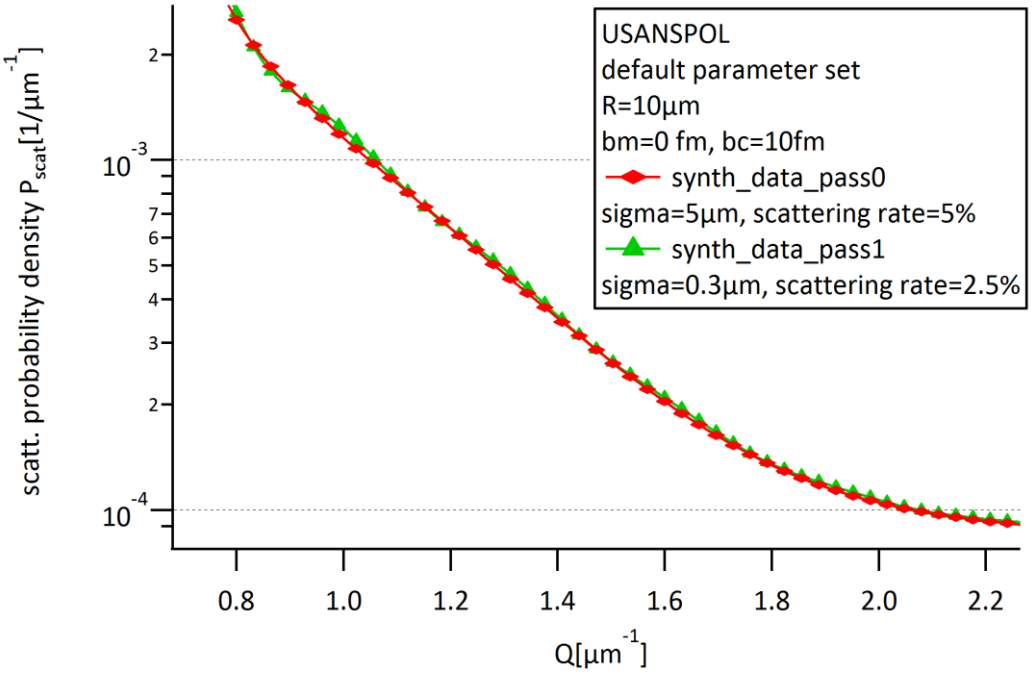

Abbildung 5.7: Detailansicht der USANSPOL Streukurve aus Abbildung 5.6. Bei einem Objektradius von R = 10 µm sind die Kurvenverläufe auch bei vergrößerter Ansicht kaum zu unterscheiden.

Es zeigt sich, dass bei Streukurven von größeren Objektradien, deren Kurvenverläufe zunehmend glatter erscheinen, eine (optische) Unterscheidung, ob es sich um geringeren Streuanteil oder ausgeprägte Objektgrößenverteilung handelt, schwer möglich ist.

5.4 USANSPOL – Unterscheidbarkeit von Objektgrößen

Ein weiteres Beispiel soll die Unterscheidbarkeit von USANSPOL Streukurven in Bezug auf unterschiedliche Objektgrößen und Streuwahrscheinlichkeiten beleuchten. Die Erfahrung aus vorigen Berechnungen lässt vermuten, dass sich die optische Unterscheidbarkeit von Kurven mit steigendem Objektradius zunehmend schwieriger gestaltet, und dass eine Erhöhung des Streuanteils in der Gesamtkurve diesem Effekt entgegenwirkt. Ausgangpunkt sei wieder eine Simulation mit dem „Default Parameter Set“, wobei der Radius R variiert wird:

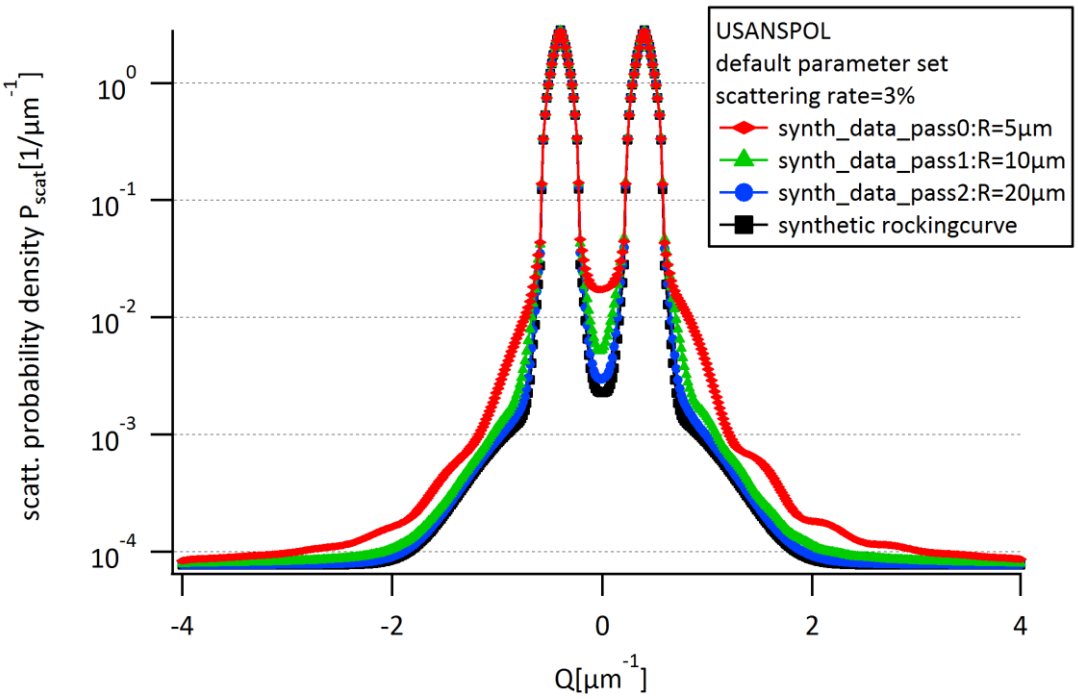

Abbildung 5.8: Synthetische USANPOL Streukurven (Default Parameter Set) von unterschiedlichen Objektgrößen im Vergleich zur Rockingkurve. Mit zunehmendem Objektradius wird die Unterscheidbarkeit der Streukurven schwieriger, der Kurvenverlauf nähert sich kontinuierlich der Rockingkurve an.

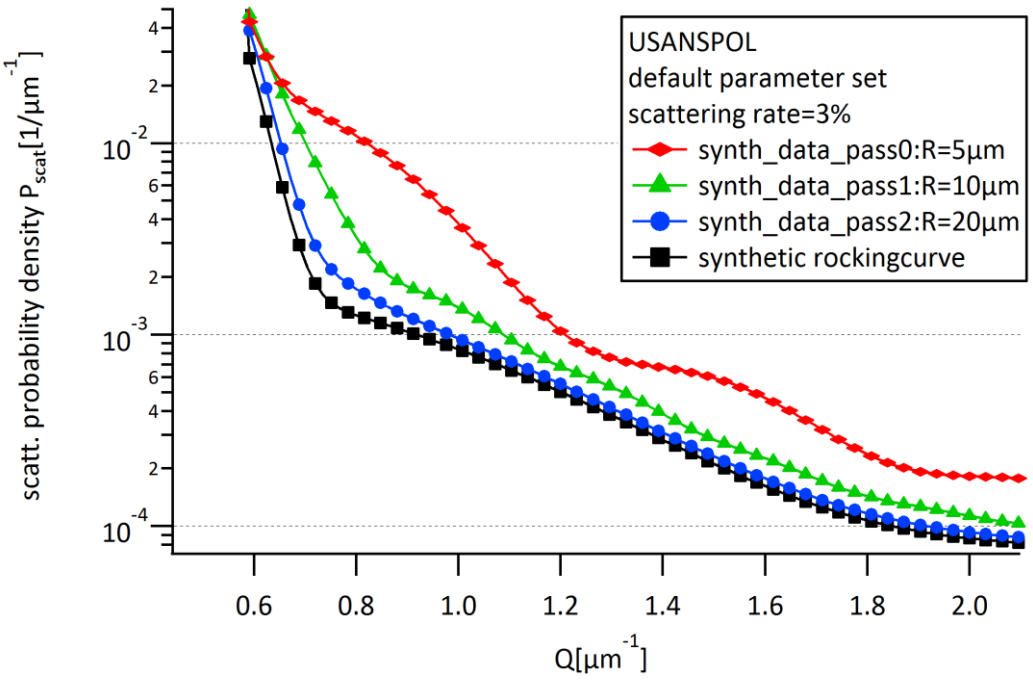

Abbildung 5.9: Detailansicht der USANSPOL Streukurve aus Abbildung 5.8. Mit steigender Objektgröße nähern sich die Kurven immer mehr der Rockingkurve an, ihre Unterscheidung gestaltet sich schwierig.

Eine optische Begutachtung von Abbildung 5.8 und im Speziellen die Detailansicht in Abbildung 5.9 zeigt, dass sich im gewählten Beispiel die Unterscheidbarkeit von Streukurven bei einem Streuanteil von 3% ab einem Radius von etwa 10 µm schwierig gestaltet. Verantwortlich dafür ist die zunehmende Annäherung der Streukurve an die Rockingkurve wenn sich die Ausdehnung der beobachteten Streuobjekte erhöht. Es liegt weiters die Vermutung nahe, dass durch Erhöhung der Streuwahrscheinlichkeit die Streukurven mehr von ihrem charakteristischen Verlauf behalten und dadurch besser optisch unterscheidbar bleiben.

Um dieser Annahme auf den Grund zu gehen, wird in der folgenden Simulation der Streuanteil auf p=10% erhöht, und der Radius R wieder im selben Bereich variiert. Die Resultate dieser Berechnung werden in Abbildung 5.10 und im Detail in Abbildung 5.11 dargestellt. Gerade aus der Detailansicht lässt sich gut die Tendenz ablesen, wie die Streukurven auf die veränderte Streuwahrscheinlichkeit reagieren: Die Abstände der Kurven untereinander haben sich deutlich vergrößert und am Beispiel von R=10 µm ist zu erkennen, dass die strukturellen Einflüsse im Kurvenverlauf besser erhalten bleiben.

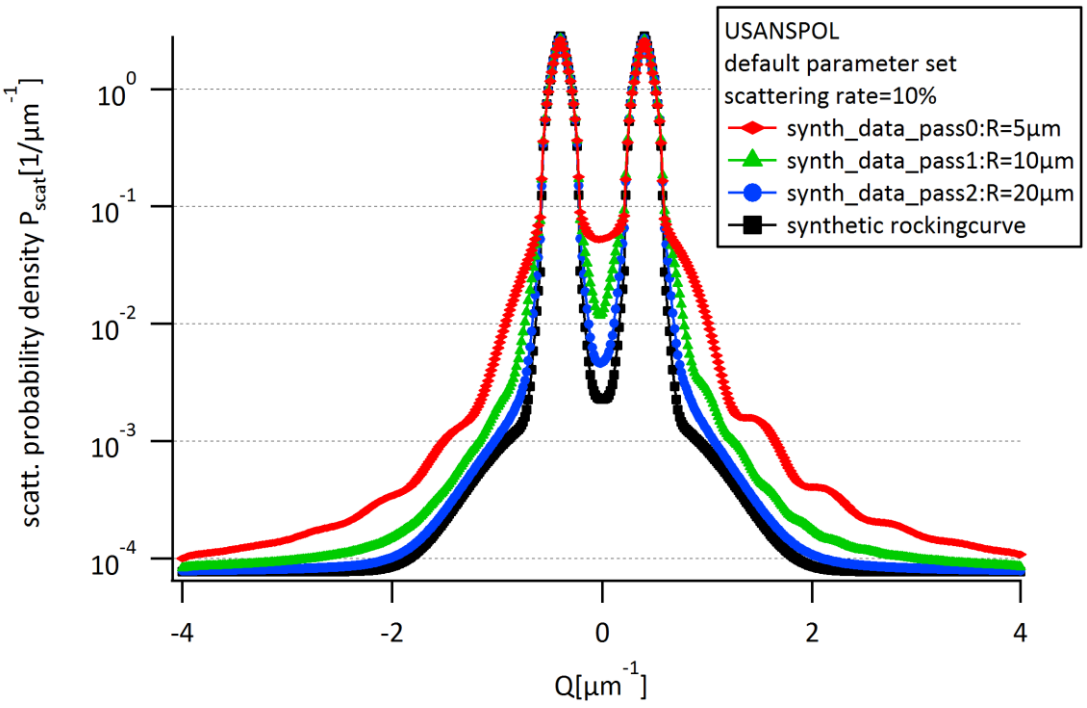

Abbildung 5.10: USANPOL Streukurven von verschiedenen Objektgrößen, Ausgangspunkt Default Parameter Set, synthetische Rockingkurve. Durch den erhöhten Streuanteil sind die Streukurven bei zunehmendem Objektradius besser unterscheidbar als in Abbildung 5.8.

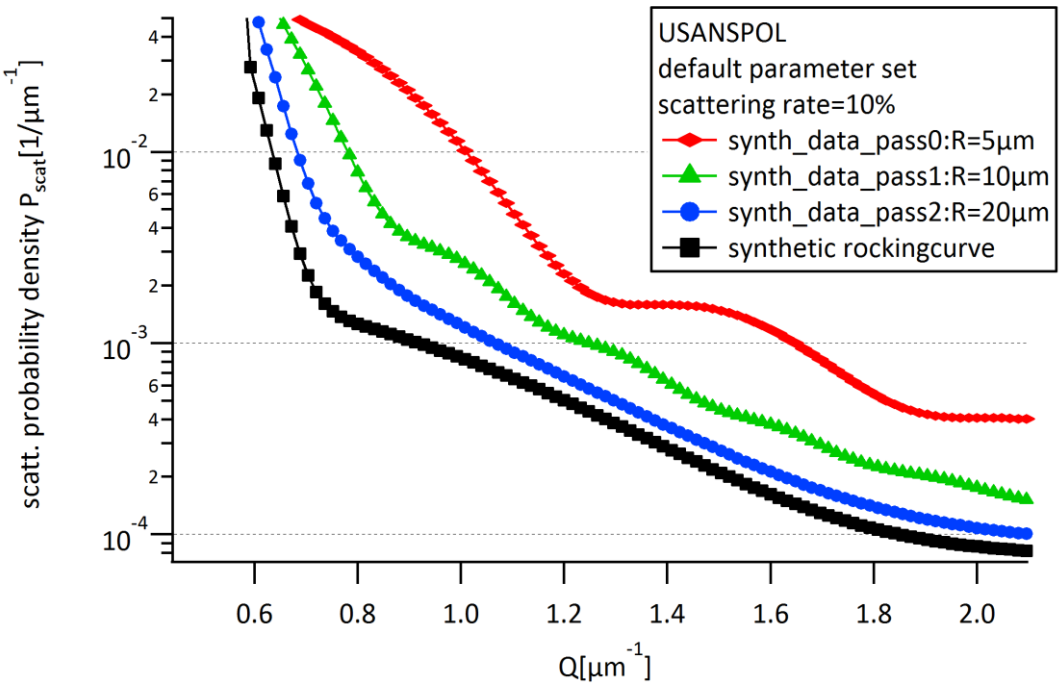

Abbildung 5.11: Detailansicht der USANSPOL Streukurve aus Abbildung 5.10.

Die in diesem Abschnitt präsentierten Auswertungen unterstreichen somit die Annahme, dass durch Erhöhung des Streuanteils in der Gesamtkurve die optische Unterscheidbarkeit von Streukurven verschiedener Objektgrößen verbessert werden kann.

5.5 USANSPOL – Streuung an magnetischen Strukturen

In diesem Beispiel soll das Verhalten der USANSPOL Streukurve während einer Variation der Parameter b_m und b_c beobachtet werden. Der Einfluss von nuklearer und magnetischer Streuung auf den Verlauf einer schlitzhöhenverschmierten Streukurve, die durch Streuung an polydispersiven Objekten entsteht, wurde bereits in Kapitel 4.2.11.1 behandelt. Im Speziellen soll veranschaulicht werden, was passiert wenn die Streuung rein magnetischen Ursprungs ist (b_c=*0*). Als Ausgangspunkt wird wieder das Default-Set an Parametern gewählt.

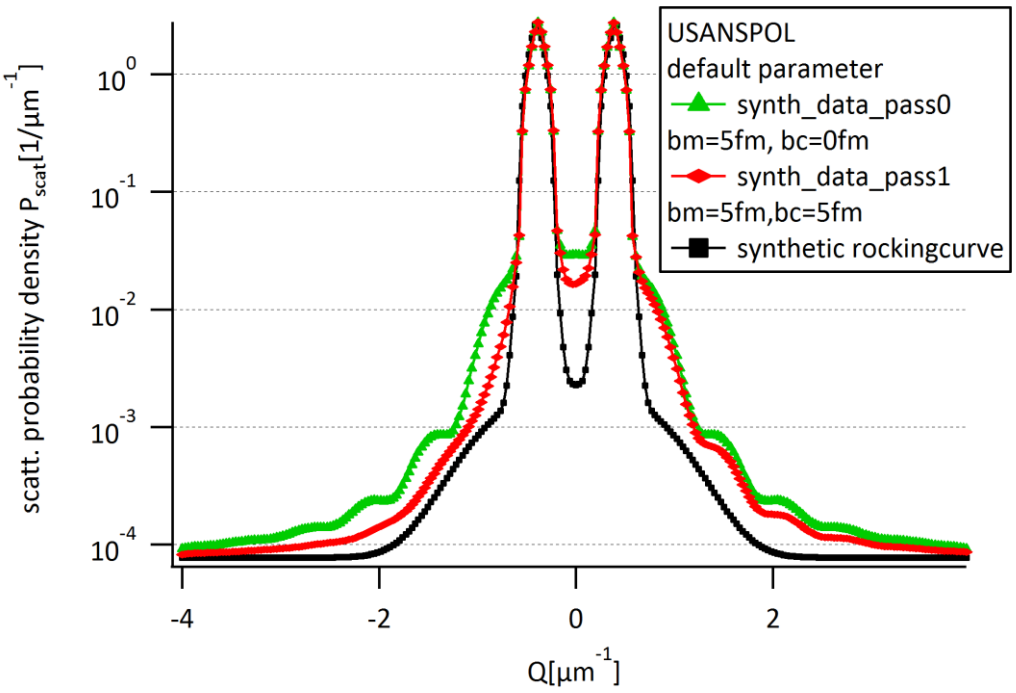

Abbildung 5.12: Vergleich von rein magnetischer und gemischter nuklearer/magnetischer Streuung. Bei b_c=0 zeigt sich die Streukurve symmetrisch, mit den charakteristischen, strukturbedingten Einflüssen an beiden Kurvenflanken. Unter Einbeziehung von nuklearer Streuung kommt es zu einem asymmetrischen Verlauf, die Spin-Down Kurvenflanke erscheint gegenüber Spin-Up abgeflacht.

5.6 USANSPOL – Einfluss der Probenorientierung

Der Einfluss der Probenorientierung auf den Verlauf einer schlitzhöhenverschmierten Streukurve, die durch Streuung an polydispersiven Objekten entsteht, wurde bereits in Kapitel 4.2.11.2 behandelt. Analog dazu soll das Verhalten der kompletten USANSPOL Streukurve unter Veränderung der Probenwinkels untersucht werden. Betrachtet man Abbildung 4.33, so lässt sich für einen Probenwinkel von π/2 ein deutlich abgeflachter Verlauf erwarten. Die nachstehende Auswertung basiert wieder auf dem „Default Parameter Set“, davon abweichend wurde b_c = 1 fm gewählt um der magnetischen Komponente, die primär von der Polarisierung und dem Magnetfeld abhängt, in der Gesamtkurve mehr Ausdruck zu verleihen. Polarisierung und Magnetisierung wurden wäh-

rend der Berechnung für alle Probenwinkel auf 0° fixiert. Das Resultat in Abbildung 5.13 zeichnet ein bekanntes Bild: Steht der Probenwinkel normal auf die Richtung von *P* und *M*, ergibt sich trotz identischer Probenstruktur ein deutlich abgeflachter Verlauf.

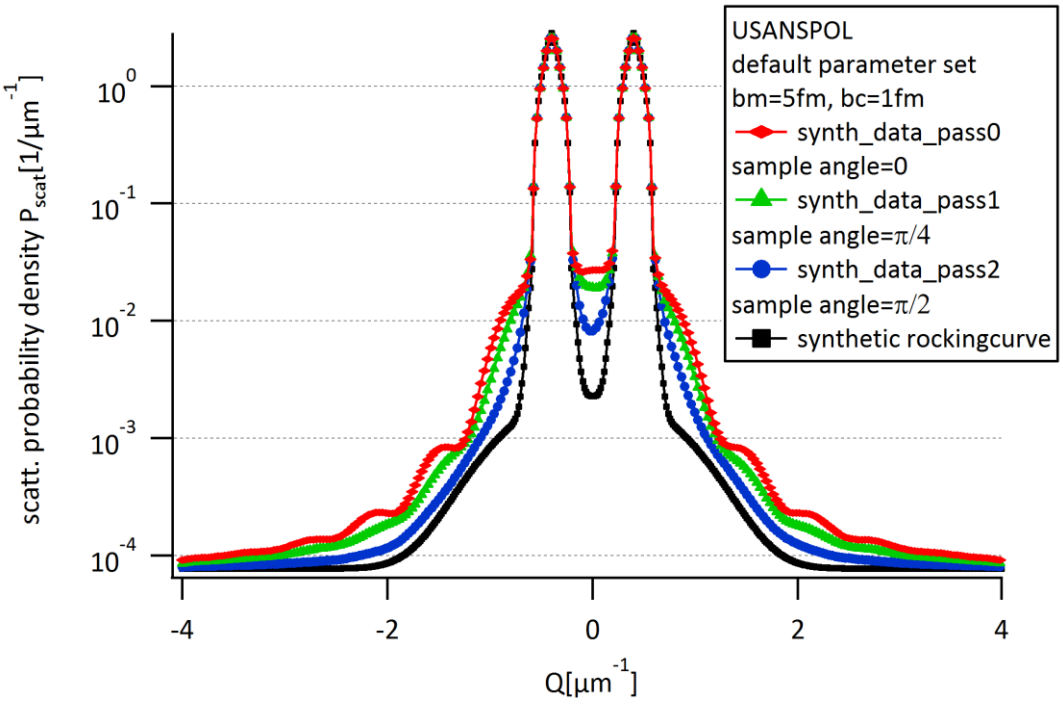

Abbildung 5.13: Synthetische USANSPOL Streukurven auf Basis Default Parameter Set bei unterschiedlichen Probenwinkeln im Vergleich zur Rockingkurve. Trotz identischer Probenstruktur zeigt sich speziell bei π/2 ein deutlich abgeflachter Verlauf der Streukurve.

6 Zusammenfassung

In dieser Arbeit wurde eine Methode beschrieben, wie durch den Einsatz von computergestützter Datensynthese aus der zugrundeliegenden mathematischen Theorie Erfahrungen im Bereich der magnetischen Ultrakleinwinkelneutronenstreuung gewonnen werden können. Bei der Erarbeitung des mathematischen Fundaments, das die Basis für diesen Ansatz bildet, wurde versucht, einen möglichst durchgängigen Formalismus zu finden und fehlende Übereinstimmungen, die bei der Literaturrecherche in kleinem Rahmen immer wieder aufgefallen sind, auszugleichen. Für die Umsetzung der Datensynthese wurde mit IGOR Pro eine durchaus leistungsstarke Plattform gefunden, die ihre Stärken vor allem im Bereich der Datenerstellung sowie Verarbeitung (Wave-Modell) und dem Bearbeiten von graphischen Darstellungen ausspielt. Deutliche Defizite im Vergleich zu anderen Softwarelösungen treten bei der Erstellung eines Userinterfaces auf, trotzdem konnte mit einigem Aufwand eine aufgeräumte und übersichtliche Oberfläche für die USANS Simulation Application bereitgestellt werden. Die einzelnen Ausbaustufen, von der Berechnung der einfachen unverschmierten Streukurve sphärischer Streuobjekte, bis hin zur magnetischen USANSPOL Kurve wurden möglichst modular aufgebaut und in die Applikation integriert. Essentielle interne Programmabläufe wurden in dieser Arbeit dokumentiert und können im Detail den angefügten Source Codes entnommen werden. Ausgehend von einem Anfangskonzept wurden im Zuge der Entwicklung viele neue Ideen geboren, die schrittweise in die Applikation implementiert wurden; so zum Beispiel wurden Möglichkeiten geschaffen, Streukurven mit Gaußschem Rauschen zu versehen, oder die Ausrichtung der spinabhängigen Peaks der USANSPOL Streukurve zu wechseln. Bei der Erstellung von Anwendungsbeispielen, mit denen gleichzeitig versucht wurde, verschiedene Fragenstellungen aus der Streudatenanalyse zu beantworten, wurden die Vorteile durch den Einsatz der Softwarelösung schnell sichtbar: Unterschiedlich parametrisierte Streukurven konnten trotz der beachtlichen Komplexität ihres Berechnungsprozesses rasch und unkompliziert erstellt werden. Es konnte gezeigt werden, dass sich die Auswirkungen von Verschmierungseffekten, die durch Polydispersivität oder Streuwahrscheinlichkeit zustande kommen, überlagern, bzw. nicht deutlich unterscheidbar sind. Simulationen mit verschiedenen Objektgrößen haben verdeutlicht, dass das Unterscheidungsvermögen von Objektgrößen im oberen Auflösungsbereich

von USANS stark von der Streuwahrscheinlichkeit beeinflusst wird. USANSPOL ist ein hervorragendes Verfahren um Untersuchungen an magnetischen Mikrostrukturen durchzuführen, aus diesem Grund werden am Atominstitut im Moment verschiedene technische Weiterentwicklungen am Experiment-Setup, speziell was die Probenumgebung betrifft, unternommen [20]. Die in diesem Zusammenhang zusätzlich benötigten Parameter wurden bereits in die Applikation integriert und ermöglichen somit die Berechnung von synthetischen Streudaten für den Fall von magnetischen Rahmenbedingungen. Daraus resultierend konnten Unterschiede zwischen nuklearem und magnetischem Streuverhalten klar aufgezeigt werden.

7 Anhang

7.1 Referenzsystem

Die Entwicklung der Applikation sowie die Erstellung aller in dieser Arbeit präsentierten Graphiken fanden ausschließlich auf dem Referenzsystem statt, seine technische Konfiguration kann der nachstehenden Auflistung entnommen werden:

HP ProBook 4530s

CPU: Intel Core i3-2330M

RAM: 4 Gb

HDD: 320 Gb

Operating System: Windows 7 Home Premium

IGOR Pro 6.2.2.2 Academic License

7.2 Default Parameter und Wertebereiche

In nachstehender Auflistung werden in der Applikation hinterlegte Standardparameterwerte- und Bereiche im Format

[ParameterName]=[Standard-Wert][Einheit][Obergrenze/Untergrenze]

angeführt.

Qmin=-4 μm^{-1} [-0.5/-30]

Qmax=4 μm^{-1} [0.5/30]

Range resolution=250 [250/10000]

Intensity Equivalent=1e+05[100/1e+10]

R=5 µm [0.1/30]

Sigma=0.3 µm [0/30]

Scattering rate=3% [0/100]

Peak shift= 0.8μm^{-1} [0/10]

bm=5 fm [-100/100]

bc=10 fm [-100/100]

Sample angle, Beam polarization angle, Atomic magnetisation angle = 0° [0/360]

7.3 Quellcode User-Defined Procedures IGOR Pro

```
#pragma rtGlobals=1        // Use modern global access method.

// USANS SIMULATION APPLICATION - Functions and Procedures
// IGOR PRO 6.22
//
//Developed by Alexander Zdarzil, last review 02/2014

//APPLICATION INIT AND RESET

function init()

	print "---\rInitialize"

	//close graph, otherwise ---\rWarning:Cannot kill waves in use
	dowindow/K graph0

	//kill all objects in datafolder
	killdatafolder root:

	//tech variables
	variable /G wavecounter=0
	variable /G gnoise_checked=0
	variable /G flip_spin_direction=0

	checkbox check0 value=0,win=panel0
	checkbox check1 value=0,win=panel0

	//physics variables
	variable /G qmin=-4
	variable /G qmax=4
	variable /G pointsperunit=250

	variable /G r=5
	variable /G sigma=.3

	variable /G rfit=0
	variable /G wstreu=3
	variable /G precision=1
	variable /G peakshift=.4
	variable /G bc=10
	variable /G bm=5

	variable /G i_aquivalent=1e+05

	variable /G theta=0
	variable /G theta_degree=0
	variable /G beam_polarisation=0
	variable /G atomic_magnetisation=0

	variable /G My,Mz,Py,Pz

	setformula theta,"convert_radiant(theta_degree)"
	setformula My,"sin(convert_radiant(atomic_magnetisation))"
	setformula Mz,"cos(convert_radiant(atomic_magnetisation))"
	setformula Py,"sin(convert_radiant(beam_polarisation))"
	setformula Pz,"cos(convert_radiant(beam_polarisation))"
```

```
	// rocking curve: load coefficient wave for function S18fit
	loadwave /P=home /Q "rockcoef.ibw"

end

// STRING FUNCTION: INIT NEW FREE WAVE AND RETURN ASSIGNED WAVE NAME

function /S makewaves(inputnamestr,qmin,qmax,radius)

	string inputnamestr
	variable qmax
	variable qmin
	variable radius

	NVAR wavecounter=wavecounter
	NVAR pointsperunit=pointsperunit

	string wname= "synth_data_pass"+num2str(wavecounter)

	Make /N=(pointsperunit) /D $wname
	SetScale x qmin,qmax,"",$wname

	note $wname, inputnamestr+" synthesis,particle radius="+num2str(radius)+"µm"

	wavecounter+=1

	return wname
end

// UNIVERSAL GRAPHING FUNCTION

function showgraph(wname)
	string wname
	print "Showing Wave Graph: " + wname

	dowindow/F graph0
	if (V_flag)
		appendtograph $wname
		ModifyGraph mode($wname)=4
	else
		display $wname
		Legend/C/N=text0
		ModifyGraph log(left)=1
		ModifyGraph mode($wname)=4
		Label bottom "\\u#2Q[µm\\S-1\\M]"
		Label left "scatt. probability density P\Bscat\M[1/µm\S-1\M]"
	endif
end

//FUNCTION CONVERT DEGREE VALUE TO RADIANT

function convert_radiant(angle)

	variable angle
	variable retval

	if (angle==0)
		retval=0
	else
		retval=(angle*(Pi/180))
```

```
	endif

	return retval

end function

//NORM AREA OF TARGETWAVE TO VALUE 1

function donorm_unity(targetwave)

	wave targetwave
	variable normfactor=area(targetwave)

	targetwave*=1/normfactor
	print "Norming area to value:1, Wavename: "+nameofwave(targetwave)+", Factor="_
	+num2str(normfactor)

	return targetwave

end funtion

//FORM FACTOR SPHERE MODEL

threadsafe function standardsphere(x,r)
	variable x,r
	variable retval

	if (x==0)
		retval=(4*pi/3)^2*r^6
	else
		retval=(4*pi/3)^2*(3*(sin(r*x)-(r*x)*cos(r*x))/(x)^3)^2
	endif

	return retval
end

//FORM FACTOR SPHERE MODEL SAMPLE ROTATION VECTOR

threadsafe function sph_vec(qy,qz,r,theta)

	variable qy,qz,r,theta
	variable retval,Q_vec

	if (qy==0 && qz==0)
		retval=(4*pi/3)^2*r^6
	else
		Q_vec=sqrt(vectorQy(qy,qz,theta)^2+vectorQz(qy,qz,theta)^2)
		retval=(4*pi/3)^2*(3*(sin(r*Q_vec)-(r*Q_vec)*cos(r*Q_vec))/(Q_vec)^3)^2
	endif

	return retval
end

//POLYDISPERSIVITY INTEGRAL SOLVED BY MATHEMATICA -
//PRODUCES AREFACTS SOMETIMES!

threadsafe function polynom_math(x,sigma,radius)
	variable x,sigma,radius
	variable value
```

```
	if (x==0)
		value=16/9*pi^2*radius^6*_
		(1+15*sigma^2/radius^2+45*sigma^4/radius^4+15*sigma^6/radius^6)
	else
		value=(1/x^6)*8*pi^2*exp(-2*x^2*sigma^2)*(exp(2*x^2*sigma^2)*_
		(1+x^2*(radius^2+sigma^2))+(-1-4*x^4*sigma^4+x^2*(radius^2-3*sigma^2))*_
		cos(2*x*radius)-2*x*radius*(1+2*x^2*sigma^2)*sin(2*x*radius))
	endif

	return value
end

//SAMPLE ROTATION QY-VECTOR

threadsafe function vectorQy(qy,qz,theta)
	variable qy,qz,theta

	return qy*cos(theta)+qz*sin(theta)
end

//SAMPLE ROTATION QZ-VECTOR

threadsafe function vectorQz(qy,qz,theta)
	variable qy,qz,theta

	return qz*cos(theta)-qy*sin(theta)
end

//MAGNETIZATION VETOR PERPENDICULAR Y

threadsafe function M_perp_y(qy,qz,My,Mz)
	variable qy,qz,My,Mz

	variable retval

	if((qy^2+qz^2)==0)
		retval=My
	else
		retval=(-qz/(qy^2+qz^2))*(qy*Mz-qz*My)
	endif

	return retval
end

//MAGNETIZATION VETOR PERPENDICULAR Z

threadsafe function M_perp_z(qy,qz,My,Mz)
	variable qy,qz,My,Mz

	variable retval

	if((qy^2+qz^2)==0)
		retval=0
	else
		retval=(qy/(qy^2+qz^2))*(qy*Mz-qz*My)
	endif

	return retval
end
```

```
//MAGNETIC SCATTERING LENGTH FUNCTION

threadsafe function scat_len(bc,bm,qy,qz,theta,npol,Py,Pz,My,Mz)
	variable bc,bm,qy,qz,theta,npol,Py,Pz,My,Mz

	variable retval

		retval=(bc+npol*bm*_
		(Py*M_perp_y(vectorQy(qy,qz,theta),vectorQz(qy,qz,theta),My,Mz)_
		+Pz*M_perp_z(vectorQy(qy,qz,theta),vectorQz(qy,qz,theta),My,Mz)))^2

	return retval

end

//NORM AREA TARGETWAVE TO AREA SOURCEWAVE

function donorm(sourcewave,targetwave)

	wave sourcewave,targetwave
	variable normfactor=area(sourcewave)/area(targetwave)

	targetwave*=normfactor
	print "Norming to Sourcewave: "+nameofwave(targetwave)+", Factor="+num2str(normfactor)

end

//LINEAR WAVE SUM - SCATTERING RATE

function dosumrock(targetwave,peakshift)
	variable peakshift
	wave targetwave

	NVAR wstreu=wstreu

	variable wavepoints=numpnts(targetwave)
	variable scalemin=leftx(targetwave)
	variable scalemax=rightx(targetwave)

	calcrock(wavepoints,scalemin,scalemax,peakshift)
	wave rockcurve
	print "Summing Scattering Curve "+nameofwave(targetwave)+_
	" with generated RockingCurve, wstreu="+num2str(wstreu)+"%"

	donorm(rockcurve,targetwave)

	targetwave*=wstreu/100
	rockcurve*=1-(wstreu/100)
	targetwave=targetwave+rockcurve

	killwaves rockcurve

end

//MODEL FUNCTION - ROCKING CURVE, SOURCE: JERICHA.E

Function S18fit(w, xx) : FitFunc
	Wave w
	Variable xx
	Variable val
```

```
	if (xx < w[2])
		val = (1 + (xx - w[2])/w[1]) * w[4]
	else
		val = (1 - (xx - w[2])/w[1]) * w[4]
	endif
	if (val < 0)
		val = (1-w[4]) * ( (1-w[6]-w[8])*exp(-((xx-w[2])/w[5])^2) +w[6]*exp(-((xx-w[2])/w[7])^2)_
		  + w[8]*exp(-((xx-w[2])/w[9])^2) )
	 else
		val += (1-w[4]) * ( (1-w[6]-w[8])*exp(-((xx-w[2])/w[5])^2) + _
		w[6]*exp(-((xx-w[2])/w[7])^2)  + w[8]*exp(-((xx-w[2])/w[9])^2) )
	 endif
	 val *= w[0]
	 val += w[3]

	return val
End

//SYNTHETIC ROCKING CURVE CALCULATION

function calcrock(wavepoints,scalemin,scalemax,peakshift)
	variable wavepoints,scalemin,scalemax,peakshift

	wave rockcoef

	Make /N=(wavepoints) /O/D rockcurve
	//S18 is scaled to 1e-4 units!!
	Setscale x scalemin*1e-4,scalemax*1e-4,"",rockcurve
	peakshift*=1e-4

	rockcurve=s18fit(rockcoef,(x+peakshift))

	Setscale x scalemin,scalemax,"",rockcurve

	print "Generated RockingCurve: "+num2str(scalemin)+"/"+num2str(scalemax)+";"+_
	num2str(wavepoints)+" points"

	return rockcurve
end

//FUNCTION FOR ADDING GAUSSIAN NOISE TO TARGETWAVE

function addgnoise(targetwave)
	wave targetwave

	//Scale to intensity äquivalent
	NVAR i_aquivalent
	variable scalefactor=(i_aquivalent/area(targetwave))
	targetwave*=scalefactor

	targetwave=targetwave+gnoise(sqrt(targetwave[p]))

	print "Added GNOISE to wave: "+nameofwave(targetwave)+", Count aquivalent: "+_
	num2str(i_aquivalent)

	return targetwave
end

//REGION: USANSPOL SYNTHESIS
```

```
//SINGLE PEAK USANSPOL - FOR DOCUMENTATION USE

function USANSPOL_synthesis_singlepeak()

	NVAR qmin=qmin
	NVAR qmax=qmax
	NVAR r=r
	NVAR peakshift=peakshift

	string wname=makewaves("Complete synthesis simple",qmin,qmax,r)
	wave targetwave=$wname

	variable pts_res=numpnts(targetwave)

	msgbox_control(1,"Processing data:")

	variable timerrefnum
	variable etime

	timerrefnum=startMStimer

	targetwave=intalg_comb_mag_simple((x+peakshift),-1,p,pts_res)
	//intalg_comb_mag_simple_old(targetwave,1)

	etime=stopMStimer(timerrefnum)
	print "---\rUSANSPOL Synthesis Simple: Single Peak generation time:"+_
	num2str(etime/1e6)+"s"

	msgbox_control(0,"")

	//doconvolve(targetwave)

	//dosumrock(targetwave)

	//donorm_unity(targetwave)

	showgraph(wname)

end

//USANSPOL - SIMPLE VARIANT OLD

function USANSPOL_synthesis_simple_old()

	NVAR qmin=qmin
	NVAR qmax=qmax

	NVAR r=r
	NVAR sigma=sigma
	NVAR bm=bm

	NVAR peakshift=peakshift
	NVAR pointsperunit=pointsperunit

	NVAR gnoise_checked
	NVAR flip_spin_direction

	string wname=makewaves("Complete synthesis simple",qmin,qmax,r)
	wave targetwave=$wname
```

```
variable shiftpoints=trunc(peakshift/deltax(targetwave))
//variable shiftpoints=x2pnt(targetwave,(qmin+peakshift))

print "shiftpoints:"+num2str(shiftpoints)

Make /N=(pointsperunit+2*shiftpoints) /D/FREE wavepart_Nup
SetScale x (qmin-peakshift),(qmax+peakshift),"" wavepart_Nup

Make /N=(pointsperunit+2*shiftpoints) /D/FREE wavepart_Ndown
SetScale x (qmin-peakshift),(qmax+peakshift),"",wavepart_Ndown

variable pts_res=(pointsperunit+2*shiftpoints)

msgbox_control(1,"Processing data:")

variable timerrefnum
variable etime
timerrefnum=startMStimer

// Calculate peak Pol_up

if (bm==0)
      wavepart_Nup=intalg_comb_simple(x,r,p,2*pts_res)
else
      wavepart_Nup=intalg_comb_mag_simple(x,1,p,2*pts_res)
endif

etime=stopMStimer(timerrefnum)
print "USANSPOL Synthesis Simple Variant generation time(P_up):"+_
num2str(etime/1e6)+"s"

// Calculate peak Pol_down

timerrefnum=startMStimer

if (bm==0)
      wavepart_Ndown=intalg_comb_simple(x,r,(p+pts_res),2*pts_res)
else
      wavepart_Ndown=intalg_comb_mag_simple(x,-1,(p+pts_res),2*pts_res)
endif

etime=stopMStimer(timerrefnum)
print "USANSPOL Synthesis Simple Variant generation time(P_down):"+_
num2str(etime/1e6)+"s"

doconvolve(wavepart_Nup)
doconvolve(wavepart_Ndown)

dosumrock(wavepart_Nup,0)
dosumrock(wavepart_Ndown,0)

if (flip_spin_direction==1)
      targetwave=wavepart_Ndown+wavepart_Nup[p+2*shiftpoints]
else
      targetwave=wavepart_Nup+wavepart_Ndown[p+2*shiftpoints]
endif

if (gnoise_checked==1)
      addgnoise(targetwave)                  // Execute if condition is TRUE
endif
```

```
	donorm_unity(targetwave)

	msgbox_control(0,"")

	showgraph(wname)

end

//USANSPOL SIMPLE VARIANT

function USANSPOL_synthesis_simple()

	NVAR r=r
	NVAR qmin=qmin
	NVAR qmax=qmax

	NVAR peakshift=peakshift

	NVAR gnoise_checked
	NVAR flip_spin_direction

	string wname=makewaves("USANSPOL synthesis simple",qmin,qmax,r)
	wave targetwave=$wname

	Duplicate/FREE targetwave,wavepart_Nup
	Duplicate/FREE targetwave,wavepart_Ndown

	variable pts_res=numpnts(targetwave)

	variable npol=1

	if (flip_spin_direction==1)
		npol=-1
	endif

	msgbox_control(1,"Processing data:")

	variable timerrefnum
	variable etime
	timerrefnum=startMStimer

	// Calculate peak Pol_up: left shift

		wavepart_Nup=intalg_comb_mag_simple((x-peakshift),npol,p,2*pts_res)

	etime=stopMStimer(timerrefnum)
	print "---\rUSANSPOL Synthesis Simple Variant generation time(P_up):"+_
	num2str(etime/1e6)+"s"

	// Calculate peak Pol_down: right shift

	timerrefnum=startMStimer

		wavepart_Ndown=intalg_comb_mag_simple((x+peakshift),-npol,(p+pts_res),2*pts_res)

	etime=stopMStimer(timerrefnum)
	print "USANSPOL Synthesis Simple Variant generation time(P_down):"+_
	num2str(etime/1e6)+"s"
```

```
	doconvolve(wavepart_Nup)
	doconvolve(wavepart_Ndown)

	dosumrock(wavepart_Nup,peakshift)
	dosumrock(wavepart_Ndown,-peakshift)

	targetwave=wavepart_Nup+wavepart_Ndown

	if (gnoise_checked==1)
		addgnoise(targetwave)			// Execute if condition is TRUE
	endif

	donorm_unity(targetwave)

	msgbox_control(0,"")

	showgraph(wname)

end

//INTEGRATION ALGORITHM: MAGENTIC SIMPLE VARIANT

function intalg_comb_mag_simple(qy_value,npol,point_number,pts_res)

	variable qy_value,npol,point_number,pts_res

	NVAR qmin=qmin
	NVAR qmax=qmax

	NVAR r=r
	NVAR sigma=sigma

	NVAR theta=theta
	NVAR bc=bc
	NVAR bm=bm

	NVAR Py=Py
	NVAR Pz=Pz
	NVAR My=My
	NVAR Mz=Mz

	variable retval
	string processing_text

	//Anzahl der Matrixpunkte festlegen
	variable pts_helperwave_poly=250
	variable pts_helperwave_slit=1000
	//Erstellen einer temporären Hilfsmatrix für die Integration
	make/N=(pts_helperwave_poly,pts_helperwave_slit)/D/FREE wTemp
	//Skalierung der Hilfsmatrix
	setscale y, get_int_limit_mag(r, qmin),get_int_limit_mag(r, qmax),"", wTemp
	setscale x, limit((r-5*sigma),0,r),r+5*sigma,"", wTemp
	//Zuweisen der Funktionswerte
	multithread wTemp=(scat_len(bc,bm,qy_value,y,theta,npol,Py,Pz,My,Mz)*_
	sph_vec(qy_value,y,x,theta))
	//Integration Matrix-columns = Integration über Qz
	integrate/DIM=1 wTemp
	//Ergebnis mit Gaussfunktion mulitplizien
	wTemp[][pts_helperwave_slit]*=gauss(x,r,sigma)
	//Integration Matrix-rows= Integration über R
```

```
	integrate/DIM=0 wTemp
	//Matrixergebnis dem Rückgabewert zuweisen
	retval=wTemp[pts_helperwave_poly][pts_helperwave_slit]

	sprintf processing_text, "Processing data: %02g%% completed",_
	(round(point_number/pts_res*100))

	msgbox_control(2,processing_text)

	return retval

end function

//INTEGRATION ALGORITHM: MAGENTIC SIMPLE VARIANT (OLD)

function intalg_comb_mag_simple_old(targetwave,neutron_spin)

	wave targetwave
	variable neutron_spin

	NVAR qmin=qmin
	NVAR qmax=qmax

	NVAR theta=theta
	NVAR bc=bc
	NVAR bm=bm

	NVAR r=r
	NVAR sigma=sigma

	NVAR Py=Py
	NVAR Pz=Pz
	NVAR My=My
	NVAR Mz=Mz

	variable ii,iii,qy,rad

	variable pts_targetwave=numpnts(targetwave)
	variable pts_helperwave_poly=250
	variable pts_helperwave_slit=250

	make/N=(pts_helperwave_slit)/D/FREE wTemp_slit
	setscale x, qmin,qmax,"",wTemp_slit

	make /N=(pts_helperwave_poly)/D/FREE wTemp_poly
	setscale x r-5*sigma,r+5*sigma,"",wTemp_poly

	ii=0
	do
		iii=0
		qy=pnt2x(targetwave,ii)

		do
			rad=pnt2x(wTemp_poly,iii)
			multithread wTemp_slit=(scat_len(bc,bm,qy,x,theta,neutron_spin,Py,Pz,My,Mz)*_
			standardsphere(sqrt(vectorQy(qy,x,theta)^2+vectorQz(qy,x,theta)^2),rad))
			wTemp_poly[iii]=area(wTemp_slit)
			iii+=1
		while(iii<pts_helperwave_poly)
```

```
            wTemp_poly*=gauss(x,r,sigma)

            targetwave[ii]=area(wTemp_poly)
            ii+=1

        while(ii<pts_targetwave)

        donorm_unity(targetwave)

        return targetwave

end function

//INTEGRATION ALGORITHM: POLY+SLIT_H SIMPLE VARIANT

function intalg_comb_simple(qy_value,r,point_number,pts_res)

        variable qy_value, r, point_number,pts_res

        NVAR sigma=sigma

        NVAR qmin=qmin
        NVAR qmax=qmax

        variable retval
        string processing_text

        //Anzahl der Matrixpunkte festlegen
        variable pts_helperwave_poly=250
        variable pts_helperwave_slit=500
        //Erstellen einer temporären Hilfsmatrix für die Integration
        make/N=(pts_helperwave_poly,pts_helperwave_slit)/D/FREE wTemp
        //Skalierung der Hilfsmatrix
        setscale y, get_int_limit(r, qmin),get_int_limit(r, qmax),"", wTemp
        setscale x, limit((r-5*sigma),0,r),r+5*sigma,"", wTemp
        //Zuweisen der Funktionswerte
        multithread wTemp=standardsphere(sqrt(qy_value^2+y^2),x)
        //Integration Matrix-columns = Integration über Qz
        integrate/DIM=1 wTemp
        //Multipikation mit Gaussfunktion
        wTemp[][pts_helperwave_slit]*=gauss(x,r,sigma)
        //Integration Matrix-rows= Integration über R
        integrate/DIM=0 wTemp
        //Matrixergebnis dem Rückgabewert zuweisen - single value
        retval=wTemp[pts_helperwave_poly][pts_helperwave_slit]

        sprintf processing_text, "Processing data: %02g%% completed",_
         (round(point_number/pts_res*100))

        msgbox_control(2,processing_text)

        return retval

end function

//USANSPOL - COMPLEX VARIANT

function USANSPOL_synthesis_complex()

        NVAR r=r
```

```
NVAR qmin=qmin
NVAR qmax=qmax

NVAR peakshift=peakshift

NVAR gnoise_checked
NVAR flip_spin_direction

string wname=makewaves("USANSPOL synthesis complex",qmin,qmax,r)
wave targetwave=$wname

Duplicate/FREE targetwave,wavepart_Nup
Duplicate/FREE targetwave,wavepart_Ndown

variable pts_res=numpnts(targetwave)

variable npol=1

if (flip_spin_direction==1)
    npol=-1
endif

msgbox_control(1,"Processing data:")

variable timerrefnum
variable etime
timerrefnum=startMStimer

// Calculate peak Pol_up: left shift

    wavepart_Nup=intalg_comb_mag_complex((x-peakshift),npol,p,2*pts_res)

etime=stopMStimer(timerrefnum)
print "---\rUSANSPOL Synthesis Complex Variant generation time(P_up):"+_
num2str(etime/1e6)+"s"

// Calculate peak Pol_down: right shift

timerrefnum=startMStimer

    wavepart_Ndown=intalg_comb_mag_complex((x+peakshift),-npol,_
    (p+pts_res),2*pts_res)

etime=stopMStimer(timerrefnum)
print "USANSPOL Synthesis Complex Variant generation time(P_down):"+_
num2str(etime/1e6)+"s"

doconvolve(wavepart_Nup)
doconvolve(wavepart_Ndown)

dosumrock(wavepart_Nup,peakshift)
dosumrock(wavepart_Ndown,-peakshift)

targetwave=wavepart_Nup+wavepart_Ndown

if (gnoise_checked==1)
    addgnoise(targetwave)                    // Execute if condition is TRUE
endif

donorm_unity(targetwave)
```

```
	msgbox_control(0,"")

	showgraph(wname)

end

//INTEGRATIONALGORITHM MAGNETIC COMPLEX VARIANT

function intalg_comb_mag_complex(qy_value,npol, point_number,pts_res)

	variable qy_value,npol,point_number,pts_res

	NVAR r=r
	NVAR sigma=sigma

	NVAR qmin=qmin
	NVAR qmax=qmax

	NVAR theta=theta
	NVAR bc=bc
	NVAR bm=bm

	NVAR Py=Py
	NVAR Pz=Pz
	NVAR My=My
	NVAR Mz=Mz

	variable int_a=-1
	variable int_b=1
	variable stepping=.1
	variable area_value_A,area_value_B,area_ratio,count_val

	variable retval
	string processing_text

	//Anzahl der Matrixpunkte festlegen
	variable pts_helperwave_poly=250
	variable pts_helperwave_slit=500
	//Erstellen einer temporären Hilfsmatrix für die Integration
	make/N=(pts_helperwave_poly,pts_helperwave_slit)/D/FREE wTemp
	//Skalierung der Hilfsmatrix
	setscale y, int_a,int_b,"", wTemp
	setscale x, limit((r-5*sigma),0,r),r+5*sigma,"", wTemp
	//Zuweisen der Funktionswerte
	multithread wTemp=(scat_len(bc,bm,qy_value,y,theta,npol,Py,Pz,My,Mz)*_
	sph_vec(qy_value,y,x,theta))
	//Integration Matrix-columns = Integration über Qz
	integrate/DIM=1 wTemp
	//Multipikation mit Gaussfunktion
	wTemp[][pts_helperwave_slit]*=gauss(x,r,sigma)
	//Integration Matrix-rows= Integration über R
	integrate/DIM=0 wTemp
	//Matrixergebnis dem Rückgabewert zuweisen - single value
	area_value_A=wTemp[pts_helperwave_poly][pts_helperwave_slit]

	do
		if (count_val>1000)
			print "---\rWarning:Cannot find limit for point:"+_
			num2str(point_number)+"/"+num2str(pts_res)
			return 0
```

```
            endif
            //Vergrößerung des Funktionsintervalls [a,b]
            int_a-=stepping
            int_b+=stepping
            //Re-Scaling mit vergrößertem Intervall [a-stepping,b+stepping]
            setscale y, int_a,int_b,"", wTemp
            setscale x, limit((r-5*sigma),0,r),r+5*sigma,"", wTemp
            //Zuweisen der Funktionswerte
            multithread wTemp=(scat_len(bc,bm,qy_value,y,theta,npol,Py,Pz,My,Mz)*_
            sph_vec(qy_value,y,x,theta))
            //Integration Matrix-columns = Integration über Qz
            integrate/DIM=1 wTemp
            //Multipikation mit Gaussfunktion
            wTemp[][pts_helperwave_slit]*=gauss(x,r,sigma)
            //Integration Matrix-rows= Integration über R
            integrate/DIM=0 wTemp
            //Matrixergebnis dem Rückgabewert zuweisen - single value
            area_value_B=wTemp[pts_helperwave_poly][pts_helperwave_slit]
            //WaveDelta, im Verhältnis zur Ausgangswave
            area_ratio=((area_value_B-area_value_A)/area_value_A)
            //letzten Integrationswert zwischenspeichern
            area_value_A=area_value_B
            //Abbruchbedingung wenn Flächenzuwachs kleiner
        while(area_ratio>0.000001)

        sprintf processing_text, "Processing data: %02g%% completed",_
         (round(point_number/pts_res*100))

        msgbox_control(2,processing_text)

        return area_value_B

end

// SLITHEIGHTSMEARING

//VARIANT 1 - INTEGRATE1D

function slitheight_st()
        NVAR qmin=qmin
        NVAR qmax=qmax

        NVAR r=r
        NVAR sigma=sigma

        string wname=makewaves("slitheight ST",qmin,qmax,r)
        wave targetwave=$wname

        variable timerrefnum
        variable etime
        timerrefnum=startMStimer

        targetwave=intslith(x)

        etime=stopMStimer(timerrefnum)
        print "Slitheight ST generation time:"+num2str(etime/1e6)+"s"

        donorm_unity(targetwave)

        showgraph(wname)
```

```
end

function intslith(x)
	variable  x
	variable /G qqy
	qqy=x
	NVAR qmin=qmin
	NVAR qmax=qmax
	NVAR precision=precision
	NVAR r=r

	return integrate1d(funcslith,(precision*qmin),(precision*qmax))
end

function funcslith(x)
	variable x

	NVAR qqy=qqy
	NVAR r=r

	return standardsphere(sqrt(qqy^2+x^2),r)

end

//VARIANT 2 - SIMPLE ALGORITHM

function slitheight_simple_mt()

	NVAR qmin=qmin
	NVAR qmax=qmax
	NVAR r=r
	NVAR pointsperunit=pointsperunit
	NVAR gnoise_checked=gnoise_checked

	string wname=makewaves("Slitheight Simple MT",qmin,qmax,r)
	wave targetwave=$wname

	variable timerrefnum
	variable etime
	timerrefnum=startMStimer

	//Punkteanzahl festlegen, targetwave=Anzahl der Punkte im Graphen
	variable pts_result=numpnts(targetwave)
	//Punkteanzahl festlegen, helperwave=Anzahl der Punkte für die Integration
	variable pts_helperwave_slit=1000
	//Erstellen eines temporären 2D wave-objekts (Matrix)
	make/N=(pts_result,pts_helperwave_slit)/D/FREE wTemp
	//Skalierung der Matrix
	setscale x, (qmin),(qmax),"",wTemp
	setscale y, get_int_limit(r, qmin),get_int_limit(r, qmax),"", wTemp
	//Berechnung der Funktionswerte in der Matrix
	multithread wTemp=standardsphere(sqrt(y^2+x^2),r)
	//Integration über dy (Columns)
	integrate/DIM=1 wTemp
	//Ergebnis aus der Matrix der Zielwave zuweisen
	targetwave=wTemp[p][pts_helperwave_slit-1]

	etime=stopMStimer(timerrefnum)
	print "---\rSlitheight simple MT generation time:"+num2str(etime/1e6)+"s"
```

```
	if (gnoise_checked==1)
		addgnoise(targetwave)		// Execute if condition is TRUE
	endif

	donorm_unity(targetwave)

	showgraph(wname)

end

//INT LIMIT FUNCTION FOR SLITHEIGHT SMEARING
//DEPENDING ON R-RANGE

function get_int_limit(radius, q_value)

	variable radius, q_value
	variable retval

	if (radius >=.1 && radius<=10)
		retval=10*q_value
	elseif (radius >10 && radius<=20)
		retval=5*q_value
	elseif (radius >20 && radius<=30)
		retval=1*q_value
	endif

	return retval

end

//INT LIMIT FUNCTION FOR SLITHEIGHT SMEARING - MAGNETIC USANS
//SMALLER RANGE BECAUSE OF SCATTERING LENGHT TERM


function get_int_limit_mag(radius, q_value)

	variable radius, q_value
	variable retval

	if (radius >=.1 && radius<=10)
		retval=5*q_value
	elseif (radius >10 && radius<=20)
		retval=2*q_value
	elseif (radius >20 && radius<=30)
		retval=.5*q_value
	endif

	return retval

end

//VARIANT 2 - SIMPLE ALGORITHM(OLD)
// FOR COMPARISION

function slitheight_simple_mt_old()

	NVAR qmin=qmin
	NVAR qmax=qmax
	NVAR r=r
	NVAR pointsperunit=pointsperunit
```

```
	NVAR gnoise_checked=gnoise_checked

	string wname=makewaves("slitheight simple(mt)",qmin,qmax,r)
	wave targetwave=$wname

	variable pts_helperwave_slit=500

	make /N=(pts_helperwave_slit)/D helperwave
	setscale x (qmin*10),(qmax*10),"",helperwave

	variable ii=0
	variable qy

	variable timerrefnum
	variable etime
	timerrefnum=startMStimer

	//Schleifeninitialisierung
	do
		//Zuweisen des x-Werts der wave im Punkt ii
		qy=pnt2x(targetwave,ii)
		//Aufbau Hilfswave
		multithread helperwave=standardsphere(sqrt(qy^2+x^2),r)
		//Integralberechnung und Zuweisung
		targetwave[ii]=area(helperwave)
		//Schleifenzähler
		ii+=1
		//Abbruchbedingung
	while(ii<numpnts(targetwave))

	etime=stopMStimer(timerrefnum)
	print "Slitheight MT generation time:"+num2str(etime/1e6)+"s"

	if (gnoise_checked==1)
		addgnoise(targetwave)			// Execute if condition is TRUE
	endif

	donorm_unity(targetwave)

	showgraph(wname)

	killwaves helperwave
end

//VARIANT 3 - COMPLEX ALGORITHM OLD

function slitheight_complex_mt_old()

	NVAR qmin=qmin
	NVAR qmax=qmax
	NVAR r=r

	variable areawave1=0
	variable areawave2=0
	variable hwqmin
	variable hwqmax
	variable deltawave

	string wname=makewaves("Slitheight Complex(mt)",qmin,qmax,r)
	wave targetwave=$wname
```

```
	variable timerrefnum
	variable etime
	timerrefnum=startMStimer

	variable ii=-1

	do
		areawave1=0
		areawave2=0
		hwqmin=-1
		hwqmax=1
		deltawave=0

		ii+=1

		string hwname="hlpwave_"+num2str(ii)
		make /N=(round((hwqmax-hwqmin)*100)) /D $hwname
		setscale/I x hwqmin,hwqmax,"",$hwname
		wave helperwave= $hwname

		multithread helperwave=standardsphere(sqrt(pnt2x(targetwave,ii)^2+x^2),r)
		areawave1=area(helperwave)
		do
			hwqmin-=1
			hwqmax+=1
			redimension/E=1/N=(round((hwqmax-hwqmin)*100)) helperwave
			setscale/I x hwqmin,hwqmax,"",helperwave
			multithread helperwave=standardsphere(sqrt(pnt2x(targetwave,ii)^2+x^2),r)
			areawave2=area(helperwave)
			deltawave=((areawave2-areawave1)/areawave1)
			areawave1=areawave2
		while(deltawave>0.00001)

		targetwave[ii]=areawave2

	while(ii<(numpnts(targetwave)))

	etime=stopMStimer(timerrefnum)
	print "Slitheight MT generation time:"+num2str(etime/1e6)+"s"

	donorm_unity(targetwave)

	showgraph(wname)

	killwaves helperwave
end

//VARIANT 3 - COMPLEX ALGORITHM

function slitheight_complex_mt()

	NVAR qmin=qmin
	NVAR qmax=qmax
	NVAR r=r
	NVAR gnoise_checked=gnoise_checked

	string wname=makewaves("Slitheight complex MT)",qmin,qmax,r)
	wave targetwave=$wname

	variable pts_res=numpnts(targetwave)
```

```
	msgbox_control(1,"")

	variable timerrefnum
	variable etime
	timerrefnum=startMStimer

	targetwave=intalg_slitheight_complex(x,r,p,pts_res)

	etime=stopMStimer(timerrefnum)
	print "---\rSlitheight complex MT generation time:"+num2str(etime/1e6)+"s"

	msgbox_control(0,"")

	if (gnoise_checked==1)
		addgnoise(targetwave)		// Execute if condition is TRUE
	endif

	donorm_unity(targetwave)

	showgraph(wname)
end

//INTEGRATION ALGORITHM: SLITHEIGHT COMPLEX VARIANT

function intalg_slitheight_complex(qy_value,r,point_number,pts_res)

	variable qy_value,r,point_number,pts_res

	variable int_a=-1
	variable int_b=1
	variable stepping=.1

	variable area_value_A,area_value_B,area_ratio,count_val

	string processing_text

	//Erstellen eines temporären Wave-Objekts
	make/N=500/D/FREE wTemp_sh
	//Skalierung der temporären Wave
	setscale x int_a,int_b,"",wTemp_sh

	//Berechnung der Funktionswerte mit Multithreading
	multithread wTemp_sh=standardsphere(sqrt(qy_value^2+x^2),r)
	//Integration über den gesamten Skalierungsbereich
	area_value_A=area(wTemp_sh)

	do
		if (count_val>1000)
			print "---\rWarning:Cannot find limit for point:"+num2str(point_number)+_
			"/"+num2str(pts_res)
			return 0
		endif
		//Vergrößerung des Funktionsintervalls [a,b]
		int_a-=stepping
		int_b+=stepping
		//Re-Scaling mit vergrößertem Intervall [a-stepping,b+stepping]
		setscale x int_a,int_b,"",wTemp_sh
		//Berechnung der Funktionswerte nach Re-Scaling
		multithread wTemp_sh=standardsphere(sqrt(qy_value^2+x^2),r)
```

```
            //Integration über den gesamten Skalierungsbereich
            area_value_B=area(wTemp_sh)
            //Berechnung WaveDelta, im Verhältnis zur Ausgangswave
            area_ratio=((area_value_B-area_value_A)/area_value_A)
            //Letzten Flächenwert zwischenspeichern
            area_value_A=area_value_B
            //Abbruchbedingung wenn Flächenzuwachs kleiner 1e-6
        while(area_ratio>0.000001)

        sprintf processing_text, "Processing data: %02g%% completed",_
         (round(point_number/pts_res*100))
        msgbox_control(2,processing_text)

        return area_value_B

end

//POLYDISPERIVITY

//VARIANT 1 - MATHEMATICA POLYNOM

function poly_mathematica()

        NVAR qmin=qmin
        NVAR qmax=qmax

        NVAR r=r
        NVAR sigma=sigma

        string wname=makewaves("polydispersive mathematica",qmin,qmax,r)
        wave targetwave=$wname

        variable timerrefnum
        variable etime
        timerrefnum=startMStimer

        targetwave=polynom_math(x,sigma,r)

        etime=stopMStimer(timerrefnum)
        print "Polydispersivity(mathematica) generation time:"+num2str(etime/1e6)+"s"

        //donorm_unity(targetwave)

        showgraph(wname)

end

//VARIANT 2 - INTEGRATE1D

function poly_ST(targetwave)
        wave targetwave

        variable timerrefnum
        variable etime
        timerrefnum=startMStimer

        targetwave=intpolydispersive(x)

        etime=stopMStimer(timerrefnum)
        print "Polydispersivity(ST) generation time:"+num2str(etime/1e6)+"s"
```

```
	donorm_unity(targetwave)

end

function intpolydispersive(x)
	variable x

	variable /G q
	q=x
	NVAR r=r
	NVAR sigma=sigma

	return integrate1d(funcpolydispersive,r-5*sigma,r+5*sigma)
end

function funcpolydispersive(x)
	variable x

	NVAR q=q
	NVAR r=r
	NVAR sigma=sigma

	return (gauss(x,r,sigma)*standardsphere(q,x))
end

//VARIANTE 3 - ALGORITHM

function poly_MT()

	NVAR qmin=qmin
	NVAR qmax=qmax
	NVAR r=r
	NVAR sigma=sigma
	NVAR pointsperunit=pointsperunit
	NVAR gnoise_checked

	string wname=makewaves("Polydispersive MT",qmin,qmax,r)
	wave targetwave=$wname

	variable timerrefnum
	variable etime
	timerrefnum=startMStimer

	//Punkteanzahl festlegen, targetwave=Anzahl der Punkte im Graphen
		variable pts_targetwave=numpnts(targetwave)
	//Punkteanzahl festlegen, helperwave=Anzahl der Punkte für die Integration
		variable pts_helperwave_poly=1000
	//Erstellen eines temporären 2D wave-objekts (Matrix)
		make/N=(pts_targetwave,pts_helperwave_poly)/D/FREE wTemp
	//Skalierung der Matrix, Limit für R=0
		setscale x, qmin,qmax,"", wTemp
		setscale y,  limit((r-5*sigma),0,r),r+5*sigma,"", wTemp
	//Berechnung der Funktionswerte in der Matrix
		multithread wTemp=gauss(y,r,sigma)*standardsphere(x,y)
	//Integration über die y-Ebene(Columns)
		integrate/DIM=1 wTemp
	//Ergebnis aus der Matrix der Zielwave zuweisen
		targetwave=wTemp[p][pts_helperwave_poly]

	etime=stopMStimer(timerrefnum)
```

```
	print "Polydispersion MT generation time:"+num2str(etime/1e6)+"s"

	if (gnoise_checked==1)
		addgnoise(targetwave)			// Execute if condition is TRUE
	endif

	donorm_unity(targetwave)

	showgraph(wname)

end

//POLYDISPERSIVITY AND SLITHEIGHT COMBINED

//VARIANT 1 - INTEGRATE1D

function combined_smearing_integrate1D()

	NVAR qmin=qmin
	NVAR qmax=qmax
	NVAR r=r

	string wname=makewaves("Combined smearing(st)",qmin,qmax,r)
	wave targetwave=$wname

	variable timerrefnum
	variable etime
	timerrefnum=startMStimer

	targetwave=intpolyslith(x)

	etime=stopMStimer(timerrefnum)
	print "Polydispersivity+Slitheight generation time:"+num2str(etime/1e6)+"s"

	donorm_unity(targetwave)

	showgraph(wname)

end

function intpolyslith(x)
	variable x
	variable /G qx
	variable /G globalr
	qx=x

	NVAR sigma=sigma
	NVAR r=r

	return integrate1d(funcpolyslith2,r-5*sigma,r+5*sigma,1)
end

function funcpolyslith1(inqy)
	variable inqy

	NVAR qx=qx
	NVAR globalr=globalr
	NVAR r=r
	NVAR sigma=sigma
```

```
	return (gauss(globalr,r,sigma)*standardsphere(sqrt(inqy^2+qx^2),globalr))
end

function funcpolyslith2(inr)
	variable inr
	NVAR globalr=globalr
	globalr=inr
	NVAR qmin=qmin
	NVAR qmax=qmax
	NVAR precision=precision

	return integrate1d(funcpolyslith1,(precision*qmin),(precision*qmax),1)
end

//VARIANT 2 - INTEGRATE1D + MATHEMATICA POLYNOM
//COMBINED

function combined_smearing_mathematica()

	NVAR qmin=qmin
	NVAR qmax=qmax
	NVAR r=r

	string wname=makewaves("Combined smearing(mt)",qmin,qmax,r)
	wave targetwave=$wname

	variable timerrefnum
	variable etime
	timerrefnum=startMStimer

	targetwave=intslithpoly(x)

	etime=stopMStimer(timerrefnum)
	print "Slitheight+Polydispersivity generation time:"+num2str(etime/1e6)+"s"

	showgraph(wname)

end

function intslithpoly(x)
	variable  x
	variable /G qy
	qy=x
	NVAR qmin=qmin
	NVAR qmax=qmax
	NVAR precision=precision

	return integrate1d(funcslithpoly,(precision*qmin),(precision*qmax))
end

function funcslithpoly(x)
	variable x
	NVAR qy=qy
	NVAR r=r
	NVAR sigma=sigma
	return polynom_math(sqrt(qy^2+x^2),sigma,r)
end

//VARIANT 3 - SIMPLE ALGORITHM
```

```
function combined_smearing_simple_mt()

	NVAR qmin=qmin
	NVAR qmax=qmax
	NVAR r=r
	NVAR gnoise_checked

	string wname=makewaves("Combined smearing simple MT",qmin,qmax,r)
	wave targetwave=$wname

	variable pts_res=numpnts(targetwave)

	msgbox_control(1,"")

	variable timerrefnum
	variable etime
	timerrefnum=startMStimer

	targetwave=intalg_comb_simple(x,r,p,pts_res)

	etime=stopMStimer(timerrefnum)
	print "---\rCombined smearing simple MT generation time:"+num2str(etime/1e6)+"s"

	msgbox_control(0,"")

	if (gnoise_checked==1)
		addgnoise(targetwave)			// Execute if condition is TRUE
	endif

	donorm_unity(targetwave)

	showgraph(wname)

end

//VARIANT 4 - ALGORITHM COMPLEX

function combined_smearing_complex_mt()

	NVAR qmin=qmin
	NVAR qmax=qmax
	NVAR r=r
	NVAR gnoise_checked=gnoise_checked

	string wname=makewaves("Combined Smearing Complex MT",qmin,qmax,r)
	wave targetwave=$wname

	variable pts_res=numpnts(targetwave)

	msgbox_control(1,"Processing data: 0% completed")

	variable timerrefnum
	variable etime
	timerrefnum=startMStimer

	targetwave=intalg_comb_complex(x,r,p,pts_res)

	etime=stopMStimer(timerrefnum)
	print "---\rCombined Smearing Complex MT generation time:"+num2str(etime/1e6)+"s"
```

```
	msgbox_control(0,"")

	if (gnoise_checked==1)
		addgnoise(targetwave)		// Execute if condition is TRUE
	endif

	donorm_unity(targetwave)

	showgraph(wname)
end

// INTEGRATION ALGORITHM: COMBINED SMEARING COMPLEX

function intalg_comb_complex(qy_value,r,point_number,pts_res)

	variable qy_value, r, point_number,pts_res

	NVAR sigma=sigma

	NVAR qmin=qmin
	NVAR qmax=qmax

	variable int_a=-1
	variable int_b=1
	variable stepping=.1
	variable area_value_A,area_value_B,area_ratio,count_val

	variable retval
	string processing_text

	//Anzahl der Matrixpunkte festlegen
	variable pts_helperwave_poly=250
	variable pts_helperwave_slit=500
	//Erstellen einer temporären Hilfsmatrix für die Integration
	make/N=(pts_helperwave_poly,pts_helperwave_slit)/D/FREE wTemp
	//Skalierung der Hilfsmatrix
	setscale y, int_a,int_b,"", wTemp
	setscale x, limit((r-5*sigma),0,r),r+5*sigma,"", wTemp
	//Zuweisen der Funktionswerte
	multithread wTemp=standardsphere(sqrt(qy_value^2+y^2),x)
	//Integration Matrix-columns = Integration über Qz
	integrate/DIM=1 wTemp
	//Multipikation mit Gaussfunktion
	wTemp[][pts_helperwave_slit]*=gauss(x,r,sigma)
	//Integration Matrix-rows= Integration über R
	integrate/DIM=0 wTemp
	//Matrixergebnis dem Rückgabewert zuweisen - single value
	area_value_A=wTemp[pts_helperwave_poly][pts_helperwave_slit]

	do
		if (count_val>1000)
			print "---\rWarning:Cannot find limit for point:"+_
			num2str(point_number)+"/"+num2str(pts_res)
			return 0
		endif
		//Vergrößerung des Funktionsintervalls [a,b]
		int_a-=stepping
		int_b+=stepping
		//Re-Scaling mit vergrößertem Intervall [a-stepping,b+stepping]
		setscale y, int_a,int_b,"", wTemp
```

```
            setscale x, limit((r-5*sigma),0,r),r+5*sigma,"", wTemp
            //Zuweisen der Funktionswerte
            multithread wTemp=standardsphere(sqrt(qy_value^2+y^2),x)
            //Integration Matrix-columns = Integration über Qz
            integrate/DIM=1 wTemp
            //Multipikation mit Gaussfunktion
            wTemp[][pts_helperwave_slit]*=gauss(x,r,sigma)
            //Integration Matrix-rows= Integration über R
            integrate/DIM=0 wTemp
            //Matrixergebnis dem Rückgabewert zuweisen - single value
            area_value_B=wTemp[pts_helperwave_poly][pts_helperwave_slit]
            //WaveDelta, im Verhältnis zur Ausgangswave
            area_ratio=((area_value_B-area_value_A)/area_value_A)
            //letzten Integrationswert zwischenspeichern
            area_value_A=area_value_B
            //Abbruchbedingung wenn Flächenzuwachs kleiner
        while(area_ratio>0.000001)

        sprintf processing_text, "Processing data: %02g%% completed",_
         (round(point_number/pts_res*100))
        msgbox_control(2,processing_text)

        return area_value_B

end

//FUNCTION CONVOLVE TARGETWAVE WITH ROCKING CURVE

function doconvolve(targetwave)
        wave targetwave

        variable wavepoints=numpnts(targetwave)
        variable scalemin=leftx(targetwave)
        variable scalemax=rightx(targetwave)

        //Cackrock without peakshift for convolution
        calcrock(wavepoints,scalemin,scalemax,0)
        wave rockcurve

        //Subtract background
        variable wmin=wavemin(rockcurve)
        rockcurve-=wmin

        donorm(rockcurve,targetwave)

        Convolve/A rockcurve, targetwave

        donorm(rockcurve,targetwave)

        targetwave+=wmin

        print "Convolving "+nameofwave(targetwave)+" with generated rockingcurve"

        killwaves rockcurve
end

// COMBINED SMEARING SIMPLE TEST FUNCTION

function combined_smearing_simple_tst()
```

```
	NVAR qmin=qmin
	NVAR qmax=qmax
	NVAR r=r
	NVAR sigma=sigma
	NVAR pointsperunit=pointsperunit

	variable ii,qy
	variable pts_helperwave_poly,pts_helperwave_slit

	string wname=makewaves("Combined smearing simple(mt)",qmin,qmax,r)
	wave targetwave=$wname

	variable pts_result=numpnts(targetwave)
	pts_helperwave_poly=250
	pts_helperwave_slit=500

	make /N=(pts_helperwave_poly)/D helperwave_poly
	setscale/I x r-5*sigma,r+5*sigma,"",helperwave_poly

	make/N=(pts_helperwave_poly,pts_helperwave_slit)/D/FREE wTemp
	setscale y, (10*qmin),(10*qmax),"",wTemp
	setscale x, r-5*sigma,r+5*sigma,"", wTemp

	variable timerrefnum
	variable etime
	timerrefnum=startMStimer

	ii=0
	do
		qy=pnt2x(targetwave,ii)

		multithread wTemp=standardsphere(sqrt(qy^2+y^2),x)
		integrate/DIM=1 wTemp

		helperwave_poly=wTemp[p][pts_helperwave_slit]*gauss(x,r,sigma)

		targetwave[ii]=area(helperwave_poly)
		ii+=1

	while(ii<pts_result)

	etime=stopMStimer(timerrefnum)
	print "Combined smearing simple(mt) generation time:"+num2str(etime/1e6)+"s"

	donorm_unity(targetwave)

	showgraph(wname)

	killwaves helperwave_poly
end

//MESSAGE BOX CONTROL FUNCTION

function msgbox_control(strg_bit,msg_text)
	//strg_bit: 0=close, 1=open, 2=update text

	variable strg_bit
	string msg_text

	switch(strg_bit)    // numeric switch
```

```
            case 0:         // execute if case matches expression
                dowindow/K msg_box
                break
            case 1:         // execute if case matches expression
                execute "msg_box()"
                titlebox title0, win=msg_box, title=msg_text
                break
            case 2:         // execute if case matches expression
                titlebox title0, win=msg_box, title=msg_text
                doupdate /W=msg_box
                break
            default:        // optional default expression executed
                            // when no case matches
        endswitch

end

//CALCULATE DOUBLE PEAK ROCKING CURVE (WITHOUT NORMING)
//FOR DOCUMENTATION USE

function calcrock_double(wavepoints,qmin,qmax,peakshift)

        variable wavepoints,qmin,qmax,peakshift

        wave rockcoef

        make /N=(wavepoints)/D/O rockcurve_double
        setscale x, qmin,qmax,"",rockcurve_double

        Make /N=(wavepoints)/D/FREE rockcurve_up
        Setscale x (1e-4*(qmin+peakshift)),(1e-4*(qmax+peakshift)), "",rockcurve_up

        Make /N=(wavepoints)/D/FREE rockcurve_down
        Setscale x (1e-4*(qmin-peakshift)),(1e-4*(qmax-peakshift)), "",rockcurve_down

        rockcurve_up=s18fit(rockcoef,x)
        rockcurve_down=s18fit(rockcoef,x)

        rockcurve_double=rockcurve_down+rockcurve_up

        return rockcurve_double

end

//ANALYSIS FUNCTIONS: SIMPLE GUINIER FIT
//EMPERIMENTAL FUNCTION

function guinierfit()

        NVAR rfit=rfit
        NVAR r=r

        rfit=0

        controlinfo /W=panel0 popup0
        string selwave=S_Value

        wave loc=$selwave
        variable zero=x2pnt(loc,0)
        variable csra=zero-round(1/r^2*100)-10
```

```
	variable csrb=zero+round(1/r^2*100)+10

	print "Guiner Fit between"+num2str(csra)+";"+num2str(csrb)

	K0=0;K1=loc(0);K2=0
	CurveFit/H="1110"/NTHR=1 gauss loc[csra,csrb]

	wave w_coef
	rfit=sqrt(5)/abs(w_coef[3])
end

//FORM FUNCTION: CYLINDRIC GEOMETRY
//FOR DOCUMENTATION USE

function cylinder(u,Q,L,R)

	variable u,Q,L,R

	variable retval

	if (u==0)
		retval=1
	elseif (Q==0)
		retval=1
	else
		retval=((sin(Q*u*(L/2))/(Q*u*(L/2)))*(2*besselj(1,(Q*R*sqrt(1-u^2)))/(Q*R*sqrt(1-u^2))))^2
	endif

	return retval

end
```

```
// USANS SIMULATION APPLICATION - Internal Procedures
// contains IDE auto-generated scripts for IGOR Objects
//
// IGOR PRO 6.22
//Alexander Zdarzil, last review 11/2013

#pragma rtGlobals=1		// Use modern global access method.

#include "USANS Procedures"

Function ButtonProc_1(ba) : ButtonControl
	STRUCT WMButtonAction &ba

	switch( ba.eventCode )
		case 2: // mouse up
			// click code here
			init()
			break
	endswitch

	return 0
End

Function ButtonProc_2(ba) : ButtonControl
	STRUCT WMButtonAction &ba

	switch( ba.eventCode )
		case 2: // mouse up
			guinierfit()
			break
	endswitch

	return 0
End

Window Graph1() : Graph
	PauseUpdate; Silent 1		// building window...
	Display /W=(253.5,51.5,648,260) hw_0
EndMacro

Function ButtonProc_14(ba) : ButtonControl
	STRUCT WMButtonAction &ba

	switch( ba.eventCode )
		case 2: // mouse up
			// click code here
			slitheight_complex_mt()
			break
		case -1: // control being killed
			break
	endswitch

	return 0
End

Function ButtonProc_16(ba) : ButtonControl
	STRUCT WMButtonAction &ba
```

```
	switch( ba.eventCode )
		case 2: // mouse up
			// click code here
			slitheight_simple_mt()
			break
		case -1: // control being killed
			break
	endswitch

	return 0
End

Function ButtonProc_11(ba) : ButtonControl
	STRUCT WMButtonAction &ba

	switch( ba.eventCode )
		case 2: // mouse up
			// click code here
				USANSPOL_synthesis_simple()
			break
	endswitch

	return 0
End

Function ButtonProc_3(ba) : ButtonControl
	STRUCT WMButtonAction &ba

	switch( ba.eventCode )
		case 2: // mouse up
			// click code here
			poly_MT()
			break
		case -1: // control being killed
			break
	endswitch

	return 0
End

Function ButtonProc_4(ba) : ButtonControl
	STRUCT WMButtonAction &ba

	switch( ba.eventCode )
		case 2: // mouse up
			// click code here
			combined_smearing_simple_mt()
			break
		case -1: // control being killed
			break
	endswitch

	return 0
End

Function ButtonProc_5(ba) : ButtonControl
	STRUCT WMButtonAction &ba

	switch( ba.eventCode )
		case 2: // mouse up
```

```
                // click code here
                combined_smearing_complex_mt()
                break
            case -1: // control being killed
                break
        endswitch

        return 0
End

Window msg_box() : Panel
        PauseUpdate; Silent 1       // building window...
        NewPanel /W=(465,302,709,346) as "Computation Info"
        SetDrawLayer UserBack
        TitleBox title0,pos={15,15},size={19,13},title="itext",frame=0
EndMacro


Function SetVarProc(sva) : SetVariableControl
        STRUCT WMSetVariableAction &sva

        NVAR beam_polarisation
        NVAR atomic_magnetisation

        switch( sva.eventCode )
            case 1: // mouse up
            case 2: // Enter key
            case 3: // Live update
                beam_polarisation = sva.dval
                atomic_magnetisation= sva.dval
                //String sval = sva.sval
                break
            case -1: // control being killed
                break
        endswitch

        return 0
End

Function CheckProc(cba) : CheckBoxControl
        STRUCT WMCheckboxAction &cba

        NVAR gnoise_checked

        switch( cba.eventCode )
            case 2: // mouse up
                gnoise_checked = cba.checked
                break
            case -1: // control being killed
                break
        endswitch

        return 0
End

Function CheckProc_1(cba) : CheckBoxControl
        STRUCT WMCheckboxAction &cba

        NVAR flip_spin_direction
```

```
	switch( cba.eventCode )
		case 2: // mouse up
			flip_spin_direction = cba.checked
			break
		case -1: // control being killed
			break
	endswitch

	return 0
End

Window Table0() : Table
	PauseUpdate; Silent 1		// building window...
	Edit/W=(5.25,42.5,510,236.75) string_container
	ModifyTable format(Point)=1,width(string_container)=174
EndMacro

function /S wlist()
	string list=WaveList("Syn*",";","")
	return list
end

Function ButtonProc(ba) : ButtonControl
	STRUCT WMButtonAction &ba

	switch( ba.eventCode )
		case 2: // mouse up
			// click code here
			NVAR r=r
			NVAR qmin=qmin
			NVAR qmax=qmax

			NVAR gnoise_checked

			string wn=makewaves("Form Factor Spherical Model",qmin,qmax,r)
			wave targetwave=$wn

			variable timerrefnum
			variable etime
			timerrefnum=startMStimer

			multithread targetwave=standardsphere(x,r)

			etime=stopMStimer(timerrefnum)
			print "---\rForm Factor Spherical Model generation time: "+num2str(etime/1e6)+"s"

			if (gnoise_checked==1)
				addgnoise(targetwave)			// Execute if condition is TRUE
			endif

			donorm_unity(targetwave)

			showgraph(wn)
			break
	endswitch

	return 0
End
```

8 Literaturverzeichnis

[1]. **Suess, D.** *Amorphes weichmagnetisches FeMoB-Band.* TU Wien, 2011. private Mitteilung.

[2]. **Wikipedia, Die freie Enzyklopädie.** *Neutron.* [Online] [Zitat vom: 2. September 2013.] http://de.wikipedia.org/w/index.php?title=Neutron&oldid=119741296.

[3]. **National Institute of Standards and Technology.** *2010 CODATA recommended values.* http://physics.nist.gov/cuu/Constants/index.html.

[4]. **J. Beringer et al. (Particle Data Group).** *Physical Review D86.* 2012.

[5]. **Wikipedia, Die freie Enzyklopädie.** *Neutronenquelle.* [Online] [Zitat vom: 2. September 2013.] http://de.wikipedia.org/w/index.php?title=Neutronenquelle.

[6]. **B.T.M. Willis, C.J. Carlile.** *Experimental Neutron Scattering.* Oxford & New York : University Press, 2009.

[7]. **H.Rauch, S.Werner.** *Neutron Inferometry.* Oxford : Clarendon Press, 2000.

[8]. **M.Hainbuchner.** *Ultra-Kleinwinkelstreuung von Neutronen an strukturierten Materialien.* TU Wien, 2000. Dissertation.

[9]. **B.Hammouda.** *Probing nanoscale structures - The SANS toolbox.* http://www.ncnr.nist.gov/staff/hammouda/the_SANS_toolbox.pdf : National Institute of Standards and Technology, Center of neutron research. (Oktober 2012).

[10]. **M.Trinker.** *Neutronen-Kleinwinkelstreuung an mikro- und nanostrukturierten Materialien.* TU Wien, 2006. Dissertation.

[11]. **A.J.Jackson.** *Introduction to Small-Angle Neutron Scattering and Neutron Reflectometry.*

http://www.ncnr.nist.gov/summerschool/ss08/pdf/SANS_NR_Intro.pdf : National Institute of Standards and Technology, Center of neutron research. (Mai 2008).

[12]. **V.F.Sears.** *Neutron optics: An introduction to the theory of neutron optical phenomena and their applications.* New York : Oxford University Press, 1989.

[13]. **A.Guinier, G.Fournet.** *Small-Angle Scattering of X-Rays.* New York : John Wiley and Sons, 1955.

[14]. **G.Porod.** *Die Röntgenkleinwinkelstreuung von dichtgepackten kolloiden Systemen.* Kolloid Zeitschrift 124, 1951. 83-114.

[15]. **L.A. Feigin, D.I. Svergun.** *Structure Analysis by Small-Angle X-Ray and Neutron Scattering.* New York : Plenum Press, 1987.

[16]. **C.Ulrich.** *Chapter2:Magnetic Neutron Scattering and Spin-Polarized Neutrons.* Stuttgart : Max-Planck-Institute for solid state research, 2007. Vorlesungsskriptum "Solid State Spectroscopy II".

[17]. **M.Villa.** *Optimiertes Kristalldesign für ein Doppelkristall-Diffraktometer.* TU Wien, 2001. Dissertation.

[18]. **Wavemetrics, inc.** *IGOR Pro Version 6.2.* Manual Revision: April 21,2011 (6.22).

[19]. **E. Jericha et al.** *Experimental and methodic progress in ultra-small-angle polarised neutron scattering on novel magnetic materials.* Journal of Physics : Conference Series 340, 2012. 012007.

[20]. **E.Jericha, G.Badurek, C.Gösselsberger.** *Towards a modelling of USANSPOL intensities from magnetic ribbons.* Phys.Procedia 42, 2013. S. 58-65.

[21]. **T.Rechberger.** *Untersuchung von Ultrakleinwinkelstreuung an magnetischen Mikrostrukturen mit polarisierten Neutronen.* TU Wien, 2013. Diplomarbeit.

[22]. **W.Mach.** *Modellunabhängige Datenanalyse von Ultrakleinwinkelstreuung mit polarisierten Neutronen.* TU Wien, 2013. Masterarbeit.